

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE SOFTWARE

TRABAJO DE INTEGRACIÓN CURRICULAR

TEMA

"DESARROLLO DE UNA APLICACIÓN WEB PARA LA DETECCIÓN DE ENFERMEDADES EN CULTIVOS DE MAÍZ MEDIANTE ARQUITECTURAS HÍBRIDAS DE DEEP LEARNING"

Trabajo de titulación previo a la obtención del título en Ingeniero en software

Línea de investigación: Desarrollo, aplicación de software y Cyber Security (seguridad cibernética)

AUTOR:

Alex Ronaldo Luna Quilumba

DIRECTOR:

PhD. Marco Remigio Pusdá Chulde

Ibarra-Ecuador (2026)

UNIVERSIDAD TÉCNICA DEL NORTE
DIRECCIÓN DE BIBLIOTECA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO	
APELLIDOS Y NOMBRES:	Luna Quilumba Alex Ronaldo

DATOS DE LA OBRA	
TÍTULO:	Desarrollo de una aplicación web para la detección de enfermedades en cultivos de maíz mediante arquitecturas híbridas de Deep learning.
AUTOR (ES):	Luna Quilumba Alex Ronaldo
FECHA: DD/MM/AAAA	19/11/2026
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniero en Software
DIRECTOR /ASESOR:	PhD. Marco Remigio Pusdá Chulde

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 19 días del mes de noviembre de 2026

EL AUTOR:

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
ALEX RONALDO LUNA QUILUMBA

.....

Nombre: Alex Ronaldo Luna Quilumba

CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 19 de noviembre de 2026

PhD. Marco Remigio Pusdá Chulde

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:

Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

MARCO REMIGIO PUSDA CHULDE

Firmado digitalmente por MARCO REMIGIO PUSDA CHULDE
Fecha: 2026.11.19 15:11:35 -05'00'

PhD. Marco Remigio Pusdá Chulde

Director de Trabajo de Grado

ÍNDICE DE CONTENIDOS

CAPÍTULO I 1
1. INTRODUCCIÓN 1
1.1 Problema de investigación 1
1.2 Objetivos 1
1.2.1 Objetivo General 1
1.2.2 Objetivos Específicos 2
1.3 Alcance 2
1.4 Metodología 3
1.5 Justificación 4
CAPÍTULO II 5
2. MARCO TEÓRICO 5
2.1 Agricultura en el mundo digital: Fundamentos 5
2.1.1 Agricultura de precisión (AP) 5
2.1.2 La agricultura 4.0 como marco de desarrollo 5
2.1.3 Desafíos y estrategias de adopción de la AP 5
2.2 Enfermedades en maíz 6
2.2.1 Vulnerabilidad y patologías comunes 6
2.2.2 Limitaciones del diagnóstico tradicional 8
2.2.3 La detección de enfermedades mediante inteligencia artificial 9
2.3 Arquitecturas de Deep Learning: Enfoques híbridos y optimización liviana 9
2.3.1 Modelos Híbridos CNN-Transformers para detección agrícola 9
2.3.2 Arquitecturas de Deep Learning livianas y eficientes 11
2.3.3 Inteligencia Artificial Explicable (XAI) y confianza en el diagnóstico 13
2.4 UAV en la agricultura de precisión 14
2.4.1 Aplicaciones y beneficios de los UAV 15
2.4.2 Teledetección y monitoreo de cultivos 15
2.4.3 Detección de enfermedades con UAV 15
2.4.4 Desventajas y limitaciones 15
2.5 ISO/IEC 25010: Modelo de calidad del software 16
2.6 Trabajos relacionados 17
CAPÍTULO III 20
3. MATERIALES Y MÉTODOS 20
3.1 Fase 1: Comprensión del Negocio 21
3.1.1 Objetivos del negocio 22
3.1.2 Evaluación de la situación 22
3.1.3 Objetivos de minería de datos 22

3.1.4 Criterios de éxito del proyecto 23
3.2 Fase 2: Comprensión de Datos 23
3.2.1 Recolección de imágenes propias mediante UAV 23
3.2.2 Datasets públicos complementarios 25
3.2.3 Análisis de datos recolectados 27
3.3 Fase 3: Preparación de Datos 28
3.3.1 Estrategia de preparación de imágenes por fuente de recolección 28
3.3.2 Anotación de imágenes con Roboflow 30
3.3.3 Preprocesamiento, división del dataset y aumento de datos 32
3.4 Fase 4: Modelado y arquitectura de la Red 34
3.4.1 Selección y justificación de la arquitectura híbrida 34
3.4.2 Arquitectura del modelo DeepLabV3+ con encoder MiT-B0 35
3.4.3 Configuración del entorno y carga del dataset 36
3.4.4 Aumento de datos en línea 37
3.4.5 Función de pérdida híbrida 38
3.4.6 Optimización y planificador de tasa de aprendizaje 39
3.4.7 Implementación de IA Explicable (XAI) con Seg-Grad-CAM 40
3.4.8 Entrenamiento del modelo 42
3.5 Fase 5: Métricas de Evaluación 43
3.5.1 Métricas de evaluación utilizadas 43
3.5.2 Proceso de evaluación del modelo 44
3.5.3 Evaluación comparativa de las arquitecturas 45
3.5.4 Evaluación cualitativa mediante mapas de calor 45
3.6 Fase 6: Despliegue 46
3.7 Metodología Scrum para el desarrollo de la aplicación web 47
3.7.1 Definición del equipo Scrum 48
3.7.2 Fase de Pre-Juego: planificación inicial 49
3.7.3 Fase de Juego: desarrollo por sprints 61
3.7.4 Fase de Post-Juego: pruebas, ajustes y despliegue 73
CAPÍTULO IV 77
4. RESULTADOS Y ANÁLISIS 77
4.1 Resultados de la construcción del dataset 77
4.2 Evaluación del modelo de inteligencia artificial 79
4.2.1 Diseño experimental para la evaluación del modelo 79
4.2.2 Comparación de arquitecturas evaluadas 80
4.2.3 Sensibilidad del modelo a la resolución de entrada 82
4.2.4 Evaluación cuantitativa del modelo seleccionado 83

4.2.5 Análisis del proceso de aprendizaje 87
4.2.6 Evaluación por categoría de enfermedad 89
4.2.7 Interpretabilidad mediante Seg-Grad-CAM 90
4.2.8 Eficiencia computacional y viabilidad de despliegue 91
4.2.9 Comparación con el estado del arte 93
4.2.10 Discusión general y limitaciones 94
4.3 Resultados de la aplicación web 96
4.3.1 Diseño del instrumento de evaluación de la aplicación 97
4.3.2 Escala de valoración y criterios de interpretación 98
4.3.3 Correspondencia entre los indicadores evaluados y la norma ISO/IEC 25010 99
4.3.4 Perfil de los usuarios participantes en la evaluación 100
4.3.5 Valoración de la eficiencia percibida en el uso de la aplicación 100
4.3.6 Valoración de la usabilidad percibida en el uso de la aplicación 104
4.3.7 Percepciones de los usuarios y recomendaciones de mejora 108
4.3.8 Discusión integral de los resultados de la evaluación de la aplicación web 108
4.4 Despliegue de la aplicación 109
CONCLUSIONES 112
RECOMENDACIONES 114
BIBLIOGRAFÍA 115

ÍNDICE DE TABLAS

Tabla 1. Comparativa CNN, ViT y enfoques híbridos**.** 11
Tabla 2. Comparativa entre arquitecturas ligeras y pesadas. 13
Tabla 3. Fases de metodología CRISP-DM implementadas en el desarrollo del proyecto**.** 21
Tabla 4. Resumen de jornadas de recolección de imágenes. 25
Tabla 5. Resumen de imágenes recolectadas de datasets públicos**.** 27
Tabla 6. Distribución total del dataset consolidado por clase y fuente**.** 27
Tabla 7. Resumen de origen, características de captura y restricciones de uso por fuente de datos. 28
Tabla 8. Imágenes anotadas por clase en Roboflow. 32
Tabla 9. Opciones de data augmentation seleccionadas en Roboflow. 33
Tabla 10. Distribución de imágenes por subconjunto tras el aumento de datos en Roboflow. 34
Tabla 11. Configuración general de la arquitectura. 36
Tabla 12. Aumento de datos en línea con Albumentations. 38
Tabla 13. Configuración del optimizador y planificador. 40
Tabla 14. Metodologías utilizadas para el proyecto. 47
Tabla 15. Equipo Scrum del proyecto. 48
Tabla 16. Nomenclatura utilizada para la planificación Scrum. 49
Tabla 17. Requerimientos funcionales. 50
Tabla 18. Requerimientos no funcionales. 51
Tabla 19. Formato de historias de usuario. 51
Tabla 20. Escala de estimación en Story Points. 52
Tabla 21. Historia de usuario HU01. 53
Tabla 22. Historia de usuario HU02. 54
Tabla 23. Historia de usuario HU03. 54
Tabla 24. Historia de usuario HU04. 55
Tabla 25. Historia de usuario HU05. 56
Tabla 26. Historia de usuario HU06. 56
Tabla 27. Historia de usuario HU07. 57
Tabla 28. Historia de usuario HU08. 58
Tabla 29. Historia de usuario HU09. 58
Tabla 30. Historia de usuario HU10. 59
Tabla 31. Product Backlog de la aplicación web. 60
Tabla 32. Distribución general de sprints del proyecto. 62
Tabla 33. Definición de terminado aplicada durante los sprints. 62
Tabla 34. Sprint Backlog – Sprint 1. 63
Tabla 35. Revisión de criterios de aceptación – Sprint 1. 63
Tabla 36. Sprint Backlog – Sprint 2. 64
Tabla 37. Revisión de criterios de aceptación – Sprint 2. 65
Tabla 38. Sprint Backlog – Sprint 3. 65
Tabla 39. Revisión de criterios de aceptación – Sprint 3. 66
Tabla 40. Sprint Backlog – Sprint 4. 67
Tabla 41. Revisión de criterios de aceptación – Sprint 4. 68
Tabla 42. Sprint Backlog – Sprint 5. 68
Tabla 43. Revisión de criterios de aceptación – Sprint 5. 69
Tabla 44. Sprint Backlog – Sprint 6. 70
Tabla 45. Revisión de criterios de aceptación – Sprint 6. 70
Tabla 46. Sprint Backlog – Sprint 7. 71
Tabla 47. Revisión de criterios de aceptación – Sprint 7. 72
Tabla 48. Matriz de trazabilidad Scrum de la aplicación web. 73
Tabla 49. Pruebas funcionales de la aplicación web. 74

Tabla 50. Correcciones y ajustes realizados en la aplicación web. 74
Tabla 51. Resumen de trazabilidad entre requisito, historia de usuario, sprint, componente, prueba y resultado esperado. 76
Tabla 52. Resumen del dataset final construido. 78
Tabla 53. Comparación del desempeño de las arquitecturas de codificador evaluadas sobre el conjunto de validación. 80
Tabla 54. Comparación de los codificadores en validación y prueba. 81
Tabla 55. Sensibilidad del codificador a la resolución de entrada sobre el conjunto de validación. 83
Tabla 56. Métricas globales del modelo sobre los conjuntos de validación y prueba. 84
Tabla 57. Métricas de segmentación por clase sobre el conjunto de validación. 89
Tabla 58. Métricas de segmentación por clase sobre el conjunto de prueba. 90
Tabla 59. Métricas de eficiencia computacional del modelo seleccionado (MiT-B0). 92
Tabla 60. Comparación referencial del modelo propuesto frente a trabajos del estado del arte en segmentación de enfermedades del maíz con imágenes de dron. 93
Tabla 61. Tecnologías utilizadas en la aplicación web. 96
Tabla 62. Criterios de interpretación de la escala de valoración. 99
Tabla 63. Correspondencia entre los indicadores del instrumento y la norma ISO/IEC 25010. 99
Tabla 64. Valoración de la eficiencia percibida por enunciado. 101
Tabla 65. Tiempos de respuesta de la aplicación en producción 103
Tabla 66. Valoración de la usabilidad percibida por enunciado. 105
Tabla 67. Resultados de la prueba de usabilidad por tarea (n = 8 participantes). 107
Tabla 68. Componentes del despliegue en producción. 110

ÍNDICE DE FIGURAS

Figura 1. Árbol de problemas. 1
Figura 2. Diagrama propuesto para el proyecto. 3
Figura 3. Enfermedad del tizón del maíz. 7
Figura 4. Enfermedad roya del maíz. 7
Figura 5. Enfermedad mancha blanca del maíz. 8
Figura 6. Arquitectura híbrida CNN-Transformers**.** 10
Figura 7. Arquitectura XAI**.** 14
Figura 8. ISO/IEC 25010. 17
Figura 9. Vuelo de dron DJI Mini 3**.** 24
Figura 10. Imágenes capturadas por dron**.** 24
Figura 11. Toma de dron DJI Mini 3 en cultivos de Otavalo**.** 25
Figura 12. Dataset público de Kaggle**.** 26
Figura 13. Recorte de imágenes por zona de interés. 29
Figura 14. Recorte de imágenes por grids**.** 30
Figura 15. Etiquetado de tizón en Roboflow. 31
Figura 16. Estrategia de anotación para cada enfermedad**.** 32
Figura 17. Arquitectura del modelo DeepLabV3+ con encoder MiT-B0. 36
Figura 18. Esquema de funcionamiento de Seg-Grad-CAM para segmentación semántica. 42
Figura 19. Flujo del proceso de entrenamiento del modelo de segmentación semántica. 43
Figura 20. Flujo de despliegue del sistema AgroScan. 47
Figura 21. Arquitectura de la aplicación web. 61
Figura 22. Interfaz de registro e inicio de sesión de la aplicación web. 63
Figura 23. Pantallas de análisis de imágenes. 65
Figura 24. Pantallas del mapa de calor y del historial de análisis. 66
Figura 25. Pantalla de la sección informativa sobre enfermedades. 67
Figura 26. Pantalla de la página principal y del módulo de gestión de la cuenta de usuario. 69
Figura 27. Pantalla de la aplicación con el modo oscuro activado. 70
Figura 28. Pantalla del módulo de análisis de parcela completa. 71
Figura 29. Matriz de confusión normalizada por fila en validación. 85
Figura 30. Matriz de confusión del modelo seleccionado (MiT-B0) sobre el conjunto de prueba: valores absolutos y normalizados por fila. 86
Figura 31. Curvas de pérdida de entrenamiento y validación y evolución del modelo MiT-B0. 88
Figura 32. Mapas de calor Seg-Grad-CAM 91
Figura 33. Interfaz de la aplicación web con el resultado de un análisis. 97
Figura 34. Flujo y arquitectura de despliegue del sistema AgroScan. 110

RESUMEN

El maíz es un cultivo clave para la seguridad alimentaria, pero su productividad se ve amenazada por enfermedades foliares cuyo diagnóstico tradicional es lento, subjetivo y depende de personal experto. Por eso este trabajo buscó desarrollar una aplicación web que detecte esas enfermedades en el maíz mediante arquitecturas híbridas de Deep Learning. Se siguió el marco **CRISP-DM.** Primero se armó un conjunto de datos con imágenes propias tomadas con vehículos aéreos no tripulados (UAV) y con datasets públicos complementarios, que se anotaron y preprocesaron. Luego se entrenó y comparó una arquitectura de segmentación DeepLabV3+ con codificador MiT-B0, sumando inteligencia artificial explicable (Grad-CAM) para interpretar las predicciones; la aplicación web se construyó con la metodología Scrum y se evaluó con la norma ISO/IEC 25010. El modelo elegido alcanzó un mIoU de 0.8182, una exactitud de píxeles de 0.9767 y un F1-score macro de 0.8972 en validación (0.7593 de mIoU en el conjunto de prueba independiente) al segmentar tres enfermedades (tizón, roya y mancha blanca), y destacó por su alta precisión al confirmar la presencia de las patologías. En las pruebas con usuarios, la aplicación tuvo una valoración favorable en eficiencia y usabilidad. En conjunto, la integración de arquitecturas híbridas de Deep Learning, imágenes UAV e inteligencia artificial explicable resulta una herramienta precisa y accesible para el diagnóstico temprano de las enfermedades del maíz, con potencial para fortalecer la agricultura de precisión.

Palabras clave: maíz, detección de enfermedades, Deep Learning, segmentación semántica, UAV, agricultura de precisión.

ABSTRACT

Maize is a key crop for food security, but its productivity is threatened by foliar diseases whose traditional diagnosis is slow, subjective, and dependent on expert personnel. For that reason, this work set out to develop a web application that detects such diseases in maize crops through hybrid Deep Learning architectures. The CRISP-DM framework was followed. First, a dataset was built from proprietary images captured with unmanned aerial vehicles (UAV) together with complementary public datasets, which were annotated and preprocessed. Then a DeepLabV3+ segmentation architecture with a MiT-B0 encoder was trained and compared, adding explainable artificial intelligence (Grad-CAM) to interpret the predictions; the web application was built with the Scrum methodology and evaluated against the ISO/IEC 25010 standard. The selected model reached a mIoU of 0.8182, a pixel accuracy of 0.9767, and a macro F1-score of 0.8972 on validation (0.7593 mIoU on the independent test set) when segmenting three diseases (northern corn leaf blight, common rust, and white spot), and it stood out for its high precision in confirming the presence of the pathologies. In the tests with users, the application received a favorable assessment in efficiency and usability. Overall, the integration of hybrid Deep Learning architectures, UAV imagery, and explainable artificial intelligence proves to be an accurate and accessible tool for the early diagnosis of maize diseases, with the potential to strengthen precision agriculture.

Keywords: maize, disease detection, Deep Learning, semantic segmentation, UAV, precision agriculture.

LISTA DE SIGLAS

AP. Agricultura de Precisión (Precision Agriculture).

API. Interfaz de Programación de Aplicaciones (Application Programming Interface).

CNN. Red Neuronal Convolucional (Convolutional Neural Network).

CRISP-DM. Proceso Estándar Multisectorial para la Minería de Datos (Cross-Industry Standard Process for Data Mining).

GPU. Unidad de Procesamiento Gráfico (Graphics Processing Unit).

Grad-CAM. Mapa de Activación de Clases Ponderado por Gradiente (Gradient-weighted Class Activation Mapping).

IA. Inteligencia Artificial (Artificial Intelligence).

IoU. Intersección sobre Unión (Intersection over Union).

ISO. Organización Internacional de Normalización (International Organization for Standardization).

MiT. Transformador Mixto (Mix Transformer).

UAV. Vehículo Aéreo No Tripulado (Unmanned Aerial Vehicle).

ViT. Transformadores de Visión (Vision Transformers).

XAI. Inteligencia Artificial Explicable (eXplainable Artificial Intelligence).

CAPÍTULO I

1. INTRODUCCIÓN

1.1 Problema de investigación

En la provincia de Imbabura, el maíz no es solo un cultivo, es el sustent;o económico de familias, sobre todo en comunidades y parroquias como La Esperanza, cantón Ibarra [1]. En los últimos años, el clima se ha vuelto más irregular; las lluvias llegan tarde o se concentran, el calor aprieta en épocas atípicas y esas oscilaciones abren la puerta a enfermedades que debilitan la planta, manchan las hojas y reducen el llenado de grano [2]. Ante ese escenario, muchos productores siguen diagnosticando como aprendieron, basándose en una simple vista, comparando síntomas y apoyándose en la experiencia empírica [3]. Ese conocimiento es valioso en el día a día, pero cuando la señal se interpreta tarde o de forma imprecisa, se responde con más agroquímicos, suben los costos y la cosecha rinde menos en la temporada [4].

El problema central no es la falta de voluntad, sino la ausencia de herramientas prácticas y asequibles [5] que permitan detectar a tiempo qué tipo de enfermedad está presente y cómo actuar con criterio (Figura 1). Sin ese apoyo, las enfermedades avanzan sin ser identificadas con precisión, se pierde la oportunidad de control temprano y el riesgo se traslada al cultivo y al bolsillo del agricultor. Por ello, se vuelve urgente probar soluciones de diagnóstico oportuno, confiables y fáciles de usar en campo, que reduzcan aplicaciones innecesarias de agroquímicos y protejan tanto la productividad como la economía rural de Imbabura [1], [6].

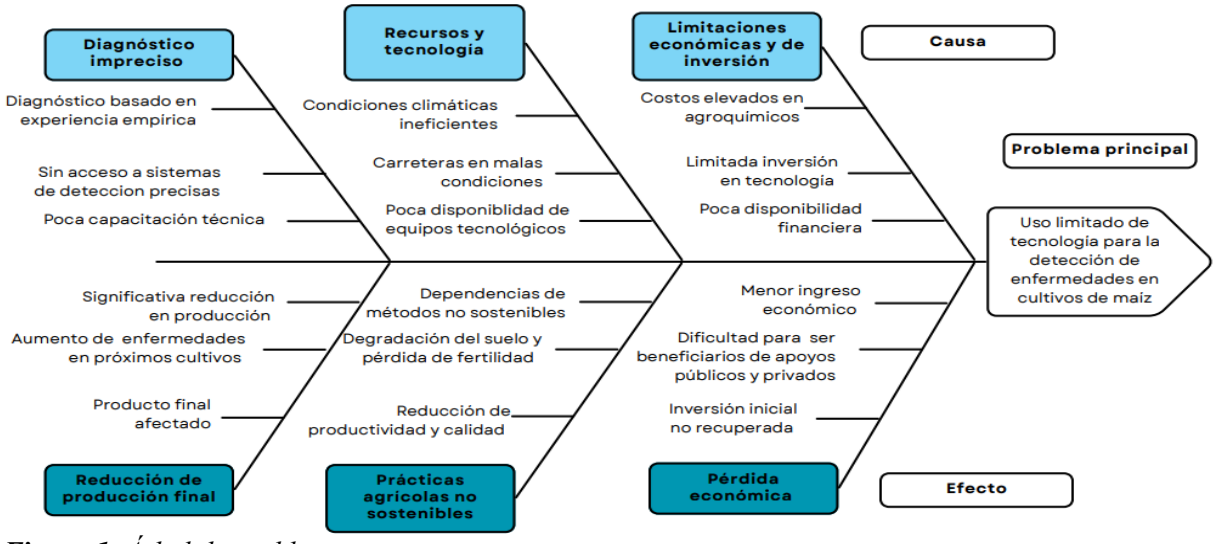


Figura 1. *Árbol de problemas.*

Fuente: Elaboración propia.

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar una aplicación web para la detección automatizada de enfermedades

en cultivos de maíz mediante arquitecturas híbridas de Deep Learning.

1.2.2 Objetivos Específicos

- Definir los fundamentos teóricos de las diferentes enfermedades que afectan al maíz y las técnicas de detección de enfermedades a través del aprendizaje profundo.
- Desarrollar una aplicación web que integre un modelo híbrido de Deep Learning basado en arquitecturas livianas que permita la detección de enfermedades del maíz.
- Evaluar el desempeño del modelo mediante métricas que permitan determinar la eficiencia y la usabilidad de la aplicación web, basadas en la norma ISO/IEC 25010.

1.3 Alcance

El alcance de este proyecto es construir una aplicación web que diagnostique las enfermedades del maíz con técnicas de Deep Learning. La idea es que el usuario pueda cargar imágenes tomadas con UAV en los cultivos y que la aplicación le señale los signos de enfermedad.

Para entrenar y validar el modelo se recolectará un conjunto de imágenes propias, tomadas con drones que llevan cámaras RGB. Las imágenes se capturan en cultivos reales de maíz, de modo que las vistas aéreas dejan ver detalles y patrones que desde el suelo cuesta notar. Ese conjunto se complementa además con el dataset público Corn Disease Drone Images de Kaggle [7] que fue tomado en un campo experimental en Nueva York.

Las imágenes propias se anotan en detalle con la herramienta Roboflow. Con ella se marcan con precisión las enfermedades presentes en cada imagen, como la mancha blanca, el tizón foliar y la roya, algo clave para que el modelo aprenda bien.

El sistema se enfoca en las enfermedades más comunes del maíz, las que dejan síntomas visibles en las hojas [2]. El proyecto usa un modelo de inteligencia artificial híbrido, apoyado en arquitecturas livianas [8], con el que se analizan y clasifican estas enfermedades. Al final, el modelo entrega resultados que pueden revisarse para tomar decisiones.

El frontend se construye con React para tener una interfaz interactiva y fácil de usar. El backend se implementa en FastAPI y se comunica de forma eficiente con el modelo de Deep Learning. El modelo se trabaja con PyTorch; el entrenamiento se hace en Google Colab y Google Drive se utiliza para el almacenamiento de los datasets ya recolectados.

Para manejar el ciclo de vida del proyecto se sigue la metodología CRISP-DM, que da un orden claro desde la investigación inicial hasta la implementación del modelo final.

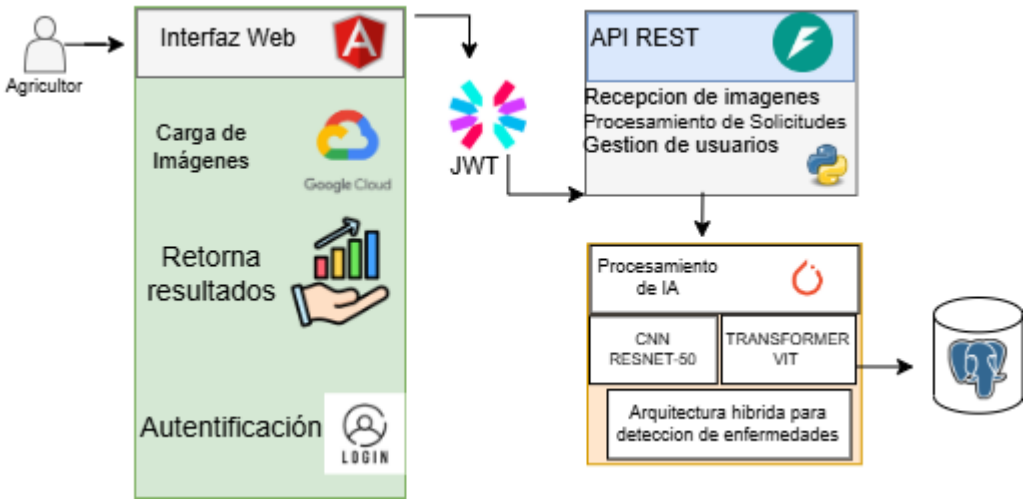


Figura 2. *Diagrama propuesto para el proyecto.*

Fuente: Elaboración propia.

1.4 Metodología

Este es un trabajo de tipo aplicado, con enfoque tecnológico y alcance experimental, ya que apunta a desarrollar una aplicación web que detecte las enfermedades del maíz a partir de imágenes capturadas por UAV. El cultivo se realiza en la comunidad de Pogllocunga, en la vía Manzano Guarangui, y en otras comunidades de la provincia de Imbabura. Todo el proyecto sigue la metodología CRISP-DM, que ayuda a ordenar por fases la investigación, el desarrollo y la evaluación, como se ve en la Figura 2.

Para el primer objetivo se hace una búsqueda a fondo de información sobre las enfermedades del maíz y sobre los usos de la inteligencia artificial en la agricultura, incluyendo sus modelos de aprendizaje y técnicas de desarrollo. Como gestor bibliográfico se usa Mendeley y se consultan fuentes académicas confiables, como revistas científicas y publicaciones especializadas.

El segundo objetivo es crear una aplicación web que use un modelo de aprendizaje profundo combinado, que aprovecha las ventajas de una arquitectura híbrida de Deep Learning y se apoya también en CRISP-DM. El modelo se entrena con un conjunto de datos propio, recolectado en campos reales de varias comunidades, y se complementa con el dataset público de Kaggle para equilibrar mejor el entrenamiento. Ya entrenado, el modelo se integra en una aplicación web hecha con React, siguiendo la metodología SCRUM.

Para el tercer objetivo se evalúa el rendimiento y la precisión del modelo con métricas cualitativas y cuantitativas, como la matriz de confusión, los mapas de calor y la precisión, entre otras. Esta evaluación sirve para confirmar qué tan fiable y eficaz es la detección de enfermedades. Además, se evalúa la usabilidad y la eficiencia del sistema web según la norma ISO/IEC 25010, de modo que el producto final cumpla con estándares de facilidad de uso y eficiencia.

1.5 Justificación

Este proyecto se alinea con varias metas de desarrollo. Aporta al ODS 2, Hambre Cero, porque detectar a tiempo las enfermedades del maíz ayuda a mejorar la producción y a cuidar la seguridad alimentaria. También se relaciona con el ODS 9, Industria, Innovación e Infraestructura, ya que una aplicación web basada en inteligencia artificial empuja la innovación tecnológica en el agro. Y encaja con el Plan de Creación de Oportunidades 2021–2025, que busca impulsar la tecnología y fortalecer al sector agrícola dentro del desarrollo productivo y sostenible del país.

Justificación tecnológica: Este sistema web es una propuesta tecnológica nueva para el sector. En el fondo lleva un modelo de aprendizaje profundo, entrenado con imágenes capturadas por UAV y procesadas con PyTorch, que identifica las distintas enfermedades del maíz con buena precisión.

Justificación social: Los cultivos de maíz se concentran sobre todo en zonas rurales, y ahí el acceso a la tecnología todavía es limitado, en especial en el norte del país. Por eso este proyecto apunta a los pequeños y medianos agricultores, y también a los técnicos agrícolas que quieren mejorar la producción y la calidad de sus cultivos. Con la aplicación se pueden detectar las enfermedades del maíz de forma temprana y precisa, sin depender de un experto, lo que facilita la toma de decisiones, el uso eficiente de los recursos y el aumento de la productividad en las comunidades rurales. Los beneficiarios directos son los agricultores y técnicos; los indirectos, sus familias, que verían una mejora en sus ingresos, y los mercados y distribuidores, que recibirían productos de mejor calidad.

Justificación agrícola: La idea es ayudar a los agricultores a sostener y aumentar sus ingresos, con la mira puesta en subir la producción final de sus cultivos. Para eso se implementa un sistema web que detecta las distintas enfermedades que afectan al maíz.

CAPÍTULO II

2. MARCO TEÓRICO

2.1 Agricultura en el mundo digital: Fundamentos

2.1.1 Agricultura de precisión (AP)

La agricultura ha experimentado un crecimiento muy significativo debido al aumento de la población mundial [9]. Sin embargo, muchas tecnologías apenas comienzan a revolucionar el ámbito agrícola. En este contexto surge la Agricultura de Precisión (AP), la cual permite satisfacer las necesidades actuales del sector, que demanda tanto la calidad como la cantidad en la producción agrícola [10].

A. Definición y enfoque

La AP se fundamenta en la capacidad de medir numéricamente variables del campo, como el estado del suelo, del aire, del agua o de la radiación, que antes solo podían estimarse a partir de la experiencia del agricultor.

B. El rol de las tecnologías de la información y comunicación (TIC)

La incorporación de las Tecnologías de la Información y la Comunicación (TIC) en la agricultura hace que los procesos sean más eficientes, reduce costos y facilita una toma de decisiones más precisa mediante el uso de plataformas digitales, Big Data, IoT y herramientas de analítica [9].

2.1.2 La agricultura 4.0 como marco de desarrollo

La AP opera dentro del contexto más amplio del enfoque de Agricultura 4.0 [9].

- **Recolección y análisis de datos:** Todos estos datos deben almacenarse, procesarse y analizarse de manera ordenada mediante técnicas de Big Data e Inteligencia Empresarial para mejorar la toma de decisiones y las acciones en el campo [9].
- **Componentes tecnológicos clave:** Todo este enfoque se lo hace a través de los componentes con los que cuenta la agricultura 4.0, como la robótica, el uso de UAV, Big Data, Inteligencia Artificial e incluso tecnologías más recientes, como las redes 5G, que en muchos países recién se han ido incorporando [9].

2.1.3 Desafíos y estrategias de adopción de la AP

A pesar de los grandes beneficios con los que cuenta la AP, su tasa de adopción aún se encuentra por debajo de las expectativas [11].

- **Alto costo de adquisición:** Se debe a que adquirir este tipo de AP implica un costo muy elevado.

- **Brecha en la adopción:** Los pocos agricultores que las utilizan son quienes cuentan con grandes plantaciones y ya tienen un ingreso más estable. Sin embargo, para agricultores promedio estas tecnologías resultan ser muy costosas y, a pesar de tener grandes beneficios, estos no se verán reflejados como un gran beneficio en plantaciones pequeñas [11].
- **Estrategia incremental:** Para evitar este tipo de inconvenientes, muchas investigaciones recomiendan un enfoque incremental utilizando solo una o dos tecnologías en cada iteración y evaluando continuamente los resultados [11].

2.2 Enfermedades en maíz

El maíz (Zea mays L.) es uno de los cereales más importantes del mundo debido a su amplia variedad de usos. En Imbabura, Ecuador, el maíz representa más que un simple cultivo; es el sustento económico de muchas familias, especialmente en parroquias rurales como La Esperanza, cantón Ibarra, donde, a pesar de la diversidad de cultivos, el maíz es su principal fuente de ingreso, por lo que estas enfermedades en los cultivos pueden afectar gravemente a la economía de los agricultores y, por ende, a la población que consume estos alimentos. [1].

2.2.1 Vulnerabilidad y patologías comunes

Este cultivo es vulnerable a múltiples enfermedades por factores como las condiciones climáticas y las prácticas agronómicas poco eficientes [2].

Condiciones climáticas: En los últimos años, los cambios climáticos se han vuelto más inestables, con lluvias tardías o concentradas y con calor en épocas atípicas. Estas oscilaciones abren la puerta a enfermedades que debilitan la planta, manchan las hojas y conllevan una producción final poco rentable para los agricultores [12].

Prácticas agronómicas poco eficientes

Las rotaciones inadecuadas, fertilización deficiente o el control insuficiente de plagas han dado como resultado el auge de varias enfermedades que afectan directamente a los cultivos del maíz [2].

Patologías comunes

Tizón de la hoja del maíz

El tizón foliar de la mancha de asfalto del norte (NCLB) ha tenido un gran impacto económico, especialmente en zonas de alta presencia de enfermedades. En los últimos años se han producido pérdidas muy significativas que van desde el 25 % hasta el 90 %, esto en Estados Unidos, el sur de

África y la India. El NCLB se manifiesta mediante lesiones en forma de puro en las hojas, que con el paso del tiempo se vuelven marrones y, posteriormente, provocan la muerte del tejido [2].

El tizón foliar de la mancha de asfalto del sur (SCLB) es más común; su presencia más activa ha sido en zonas húmedas, tropicales y templadas cálidas, llegando a dejar pérdidas del 70 %.

La semejanza entre estas dos enfermedades ha sido un gran desafío para identificarlas, incluso para expertos en el área, especialmente en su fase temprana. La Figura 3 muestra el aspecto de esta enfermedad en el cultivo de maíz.



Figura 3. *Enfermedad del tizón del maíz.*

Fuente: Elaboración propia.

Roya común

La roya común del maíz aparece por primera vez tras la seda y se muestra como pústulas alargadas de color óxido o también como un marrón oscuro en la superficie de la hoja del maíz. Su presencia es más común en América, Asia y África [2]. En la Figura 4 se observa el aspecto de esta enfermedad en la hoja de maíz.



Figura 4. *Enfermedad roya del maíz.*

Fuente: Elaboración propia.

Mancha Blanca

La mancha blanca del maíz es una de las enfermedades foliares más agresivas del cultivo en Brasil y en otros países de la región, incluido Ecuador, y puede ocasionar pérdidas de rendimiento que, en casos severos, superan el 60 %. Sus síntomas se manifiestan inicialmente como pequeñas manchas acuosas de color verde oscuro que, con el avance de la infección, evolucionan a lesiones necróticas circulares u ovaladas de color pajizo o blanquecino, con márgenes bien definidos, que reducen progresivamente el área fotosintética de la hoja. Su agente causal ha sido objeto de controversia: en un inicio la enfermedad se describió como mancha foliar de Phaeosphaeria, atribuida al hongo Phaeosphaeria maydis; no obstante, estudios más recientes identifican a la bacteria Pantoea ananatis [13] como el principal agente causal de la enfermedad. En la Figura 5 se observa el aspecto de esta enfermedad en la hoja de maíz.



Figura 5. *Enfermedad mancha blanca del maíz.*

Fuente: Elaboración propia.

2.2.2 Limitaciones del diagnóstico tradicional

A pesar de estas diversas infecciones que se dan, muchos agricultores siguen diagnosticando las enfermedades basándose en el conocimiento adquirido, una simple vista y la comparación de síntomas, lo que se conoce como la experiencia empírica [3].

Consecuencias de la imprecisión: Este conocimiento es valioso en el día a día; sin embargo, si la señal se interpreta tarde o de forma imprecisa, esto puede ocasionar un uso excesivo de agroquímicos y, por ende, mayor inversión [4].

Impacto económico y ambiental: Esto provoca un aumento de los costos y una reducción del rendimiento de la cosecha. El uso limitado de tecnología para la detección precisa de enfermedades en cultivos de maíz es el problema principal que resulta en una reducción significativa de la producción final y en una pérdida económica para el agricultor [4], [14].

2.2.3 La detección de enfermedades mediante inteligencia artificial

La inteligencia artificial (IA), mediante técnicas de aprendizaje profundo y visión artificial, ofrece una solución al diagnóstico impreciso y tardío más eficaz en comparación con los conocimientos empíricos de los agricultores [14].

Diagnóstico oportuno y confiable: La IA permite identificar con mayor precisión diversos tipos de enfermedades presentes en los cultivos.

Agricultura sostenible: Gracias al apoyo de la IA, se pueden aplicar tratamientos precisos, reducir las aplicaciones innecesarias de agroquímicos y proteger tanto la productividad como la economía rural.

Procesamiento de imágenes

El procesamiento de imágenes desempeña un papel fundamental en diversas áreas, como la detección precisa de enfermedades, frutos y malezas en los cultivos, y en muchas aplicaciones que contribuyen a la agricultura inteligente. Sin embargo, la aplicación de esta tecnología tiene varios inconvenientes como la interferencia ambiental, la calidad de los datos, desequilibrio en conjunto de datos, entre otros [15]. Para el procesamiento de imágenes, se busca obtener tomas de alta calidad de los cultivos que permitan detectar con gran precisión el estado de las plantas. Para ello se ha introducido el uso de cámaras digitales, integradas en drones o dispositivos móviles; todo esto ha facilitado la toma de imágenes de grandes extensiones en muy poco tiempo. Además, el uso de redes neuronales profundas ha sido una pieza fundamental para la clasificación de imágenes, lo que impulsa la agricultura de precisión a través de un monitoreo automático [16].

2.3 Arquitecturas de Deep Learning: Enfoques híbridos y optimización liviana

El desarrollo de un modelo de inteligencia artificial es el pilar fundamental para el funcionamiento de la aplicación web; se ha optado por utilizar un enfoque híbrido de Deep Learning; sin embargo, también buscamos utilizar arquitecturas livianas capaces de ejecutarse sin problemas en dispositivos con tecnología limitada [17].

2.3.1 Modelos Híbridos CNN-Transformers para detección agrícola

El enfoque de este proyecto es el uso de imágenes de vehículos aéreos no tripulados UAV, que tienen algunas ventajas a la hora de realizar detecciones de enfermedades especialmente en grandes extensiones, sin embargo, estas imágenes incluyen variaciones de iluminación, patrones de enfermedades que no son muy visibles [18] Es por ello por lo que se requiere un modelo que combine lo mejor de las arquitecturas convolucionales y las basadas en atención; para ello, se ha buscado desarrollar un enfoque híbrido, tal como se puede apreciar en la *Figura 6.*

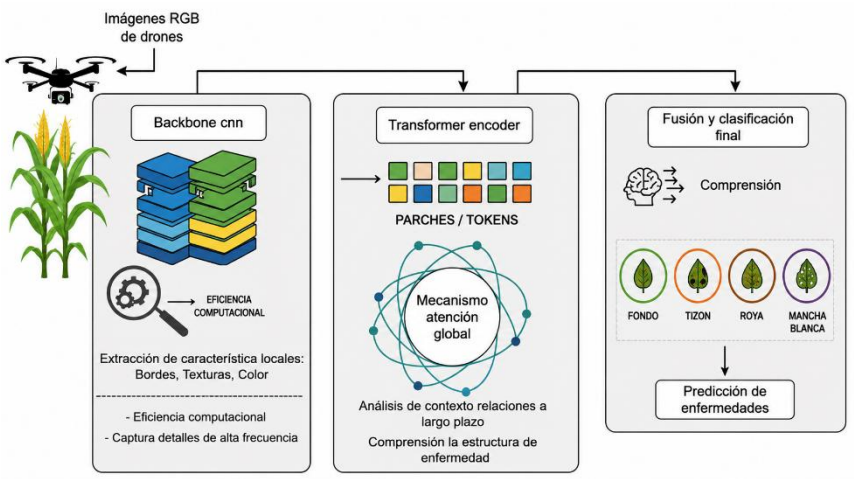


Figura 6. *Arquitectura híbrida CNN-Transformers.*

Fuente: Elaboración propia.

Limitaciones de modelos puros.

Redes Neuronales Convolucionales CNN

Estas redes son muy eficientes para analizar imágenes de hojas; tienen una gran capacidad para extraer características locales de la imagen, como bordes, textura, color y formas, para que sean clasificadas en diferentes clases según el tipo de enfermedad. Sin embargo, estas redes pueden memorizar los datos si un dataset es pequeño, lo que bajaría la precisión del modelo [18].

Vision Transformers (ViT)

Las redes neuronales tienen un enfoque diferente al de las CNN, ya que las Transformers dividen las imágenes en pequeños parches que, por lo general, son de 64 píxeles y luego aplican su mecanismo de atención en cada uno de esos parches. Esta técnica permite capturar las dependencias y relaciones de cada parche, generando así una comprensión global del síntoma que presenta la hoja , [19].

Ventajas del enfoque híbrido

El enfoque híbrido, candidato a entrenamiento del modelo para este proyecto, combina CNN y ViT para lograr una mayor precisión.

Extracción de características CNN: La primera etapa es la extracción de características locales de la imagen de entrada, que utiliza un CNN liviano, como EfficientB0, ShufflenetV2, YOLOv12n, entre otros, que reducen significativamente la dimensionalidad de los datos.

Análisis global Transformers: Una vez extraído los mapas de características de las CNN, pasan a una segunda fase en la que son aplanados y posteriormente convertidos en tokens o parches donde se aplica los mecanismos de atención de las redes Transformers donde captura las relaciones globales de la hoja para diferenciar con mayor precisión los síntomas que presenta esa imagen [18]. En la Tabla 1 se comparan las características de las CNN, los Vision Transformers y los enfoques híbridos.

Tabla 1. *Comparativa CNN, ViT y enfoques híbridos.*

Característica	**Redes convolucionales (CNN)**	**Vision Transformers (ViT)**	**Enfoque Híbrido**
Enfoque de análisis	**Local:** Su enfoque se centra en texturas, bordes o diferentes formas que se puedan encontrar en la imagen [18].	**Global:** Aplica mecanismos de atención [19].	**Integral:** Se utilizan características para convertirlas en tokens para posteriormente aplicar mecanismos de atención [20].
Mecanismos claves	Capas de convolución y pooling [18].	Mecanismos de autoatención (*Self-attention*) [20].	Convoluciones con bloques de atención de Transformers.
Limitaciones	Riesgo de sobreajuste especialmente en datasets más pequeños [18].	Necesitan mayor poder de cómputo y más cantidad de datos [20].	Necesitan una implementación adecuada para aprovechar el rendimiento de los dos enfoques [20].
Aplicación utilizando UAV.	Muy sensibles a cambios de iluminación.	Más estable con imágenes con patrones amplios.	Más eficiente y óptimo para utilización en campos reales.

Fuente: Elaboración propia.

2.3.2 Arquitecturas de Deep Learning livianas y eficientes

A pesar de un alto rendimiento y precisión de muchos modelos de Deep Learning, estos conllevan un gran costo, ya que requieren un gran poder de cómputo para funcionar correctamente. El objetivo del proyecto es ayudar a los agricultores, quienes no cuentan con muchos recursos; por ello optamos por utilizar modelos livianos y, al mismo tiempo, eficientes [21].

Estrategias de eficiencia computacional

Un modelo liviano se caracteriza por su capacidad de ejecutarse en la gran mayoría de dispositivos tecnológicos, especialmente en los de bajos recursos, todo esto se debe a que este tipo de modelos trabajan con una baja cantidad de parámetros y una baja cantidad de operaciones de punto flotante (FLOPs) [21], sus principales estrategias de optimización son:

Convoluciones separables en profundidad

Es una técnica que se basa en descomponer una CNN en dos pasos: convolución en profundidad (depthwise), que filtra cada canal de entrada por separado, y convolución puntual (pointwise), que combina la información entre todos los canales, reduciendo así el número de parámetros y operaciones [21].

Transfer Learning

Es una técnica que realiza un modelo preentrenado para mejorar el rendimiento de un nuevo problema, sin embargo, utilizará menos datos y poder de cómputo que si se entrenara desde cero [22].

Arquitecturas híbridas livianas

El modelo MobileViT (Mobile Vision Transformer) propuesto como una solución que combina la eficiencia de las redes neuronales convolucionales livianas con la capacidad de captura global de los Vision Transformers [17]

Fundamento: Esta arquitectura está diseñada para dispositivos móviles. Utiliza las convoluciones de MobileNet para la extracción de características y luego integra bloques de Vision Transformer livianos que operan sobre características locales.

Por otra parte, las arquitecturas mit [23], que cuentan con varias versiones desde el modelo mit_b0, con alrededor de 4 millones de parámetros, y modelos más pesados, como mit_b2, que pasan de 20 millones de parámetros, son arquitecturas que combinan dos mundos de Deep Learning, como Transformers, que permiten capturar relaciones globales de una imagen mediante atención, y, además, con redes neuronales convolucionales que contienen bloques Mix-FFN e incorporan convoluciones Depth-wise para conservar información espacial. En la Tabla 2 se contrastan las arquitecturas ligeras con las arquitecturas pesadas.

Tabla 2. *Comparativa entre arquitecturas ligeras y pesadas.*

Características	Arquitecturas Ligeras.	Arquitecturas Pesadas
Consumo de recursos	Se caracterizan especialmente por su reducida cantidad de parámetros y FLOPs.	Consumen un gran recurso en poder de cómputo y memoria.
Estrategia de rendimiento.	Su arquitectura se basa en convoluciones separables en profundidad.	Capas densas y convoluciones extremas.
Rendimiento de Datos	Su optimización se debe al uso de transfer learning, donde el modelo no se entrena desde cero.	Requieren una gran cantidad de datos para un entrenamiento desde cero.
Equipos de aplicación	Se limita a dispositivos con tecnología de bajos recursos.	Están hechas para uso en dispositivos de alto rendimiento.

Fuente: Elaboración propia.

2.3.3 Inteligencia Artificial Explicable (XAI) y confianza en el diagnóstico

El uso del Deep Learning en la agricultura es un enfoque muy reciente, por lo que muchos agricultores aún no confían en este tipo de herramientas. Es ahí donde aparece la IA explicable, que es capaz de mostrar los puntos en donde el modelo se fijó para dar un diagnóstico, siendo más aceptable por los agricultores [24].

Definición e importancia de XAI

La XAI resulta ser muy fundamental en ámbitos donde el uso de IA no ha sido muy aceptado por sus usuarios, este es caso de los agricultores donde, los usuarios no confían mucho en los diagnósticos que te da la IA, pues para mucho de los agricultores esta técnica no resulta confiable y por ende no lo aceptan como una herramienta de apoyo es ahí donde aparece la IA explicable que no solo muestra un diagnóstico, si no que agrega patrones y mapas de calor tal como se muestra en la Figura 7 para mostrar las áreas donde se fijó y dar un resultado, esto a

resultado bastante eficiente para los agricultores ya que les da un diagnóstico más confiable [24].

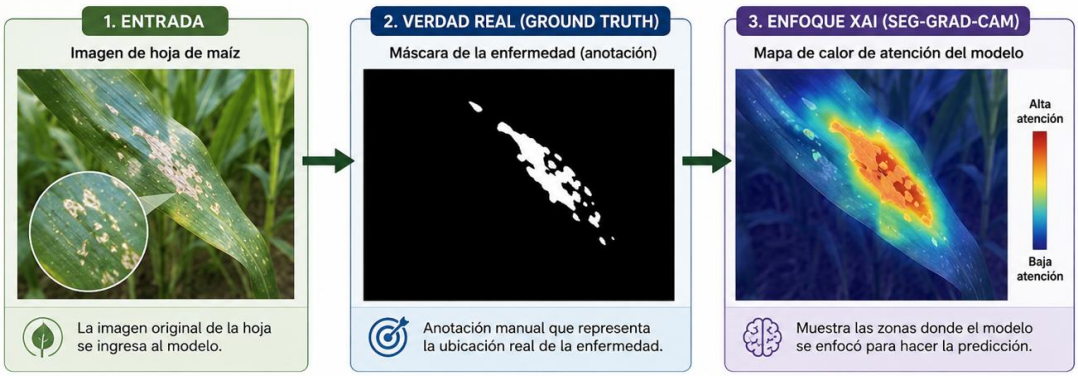


Figura 7. *Arquitectura XAI.*

Fuente: Elaboración propia.

Validación del diagnóstico

Permite al agricultor ver visualmente la decisión que ha tomado el modelo; puede ver si el modelo en verdad se centró en alguna mancha o infección de la hoja, o si escogió cualquier parte del fondo, como el suelo u otra planta cercana, y da un resultado erróneo [24].

Grad-CAM como herramienta de explicación

Gradient-weighted Class Activation Mapping (Grad-CAM) es una técnica muy usada por CNN para mostrar un mapa de calor del lugar donde se centró para dar un diagnóstico. Grad-CAM logra esto utilizando información de gradiente específica de la clase que fluye hacia la capa convolucional final de la red, lo que permite una mejor interpretación del comportamiento interno de los modelos de aprendizaje profundo [24]. Sin embargo, para crear mapas de calor más acordes con el presente proyecto, se revisó una técnica llamada Seg-Grad-CAM [25], que es como un submódulo de Grad-CAM que está más enfocado en lo que es segmentación de imágenes, que permite analizar a nivel de píxel o región.

2.4 UAV en la agricultura de precisión

La agricultura de precisión ha tenido un gran auge con el avance de la tecnología y muchas herramientas tecnológicas se han incorporado en este ámbito. Entre ellas, también ha revolucionado el uso de vehículos aéreos no tripulados (UAV) que cuentan con un amplio equipamiento tecnológico, como cámaras RGB y sensores, entre otros [26].

2.4.1 Aplicaciones y beneficios de los UAV

En los últimos años se han desarrollado UAV capaces de realizar tareas que, de forma tradicional, tomarían mucho tiempo [8] Entre las ventajas del uso de UAV tenemos:

- **Eficiencia en fumigación:** Un claro ejemplo son los UAV de fumigación, que cuentan con tecnología para cubrir grandes hectáreas en un solo vuelo, lo que, si se hace de forma tradicional, tomaría días enteros.
- **Reducción de insumos y costos:** Además, no solo pueden cubrir grandes extensiones de fumigación, sino que también reducen el consumo de agroquímicos, lo que ayuda a disminuir costos significativos para los agricultores.

2.4.2 Teledetección y monitoreo de cultivos

El uso de UAV en la agricultura abre la puerta a nuevas tecnologías, entre ellas la teledetección.

- **Ventajas en el monitoreo:** La teledetección permite monitorear los cultivos sin invadir directamente la plantación y, además, resulta más rápida y eficiente.
- **Fundamento tecnológico:** Todo esto se debe a los UAV, a los diferentes tipos de sensores con los que cuentan y, especialmente, a los algoritmos de detección de imágenes [27].
- **Superioridad frente a satélites:** Aunque también existe la teledetección satelital, la incorporación de UAV ha sido más provechosa, ya que permite obtener imágenes de mejor calidad, especialmente porque no se ve afectada por la interferencia de las condiciones atmosféricas [27].

2.4.3 Detección de enfermedades con UAV

Otra de las áreas beneficiadas por la integración de los drones es la detección de enfermedades en las plantas.

- Esto ayuda a detectar de forma temprana el tipo de enfermedad, evitando la propagación de infecciones que afecten el rendimiento y la calidad del producto final [28].

2.4.4 Desventajas y limitaciones

El uso de UAV ha sido muy beneficioso en el ámbito agrícola; sin embargo, también presenta ciertas desventajas.

- **Control y operación:** Siempre debe haber alguien a cargo del control del dron [28].

- **Limitaciones de carga y sensores:** Los sensores que se instalan no siempre serán los mejores, ya que los drones no pueden cargar cargas excesivas [28].
- **Costo:** Los costos de implementación suelen ser muy elevados, especialmente para agricultores de comunidades rurales que no cuentan con una economía sostenible significativa [1].
- **Condiciones climáticas:** Su principal desventaja podría ser el clima, ya que, al ser muy ligeros, en condiciones de viento fuerte no pueden trabajar de la mejor manera [28].

2.5 ISO/IEC 25010: Modelo de calidad del software

Los modelos de inteligencia artificial no solo necesitan una buena precisión, sino que también deben tener sus respectivos estándares internacionales de calidad, es ahí donde incorporamos la norma ISO/IEC 25010 que constituye un marco de referencia fundamental, al definir un modelo de calidad que permite evaluar productos y sistemas de software en función de características específicas [29].

Este estándar forma parte de la familia SQuaRE (Software Quality Requirements and Evaluation) [29], que establece ocho características principales y cada una de ellas se divide en subcaracterísticas que nos permiten medir de forma objetiva la calidad del software. En la Figura 8 se presentan las características y subcaracterísticas de calidad que establece la norma.

En el contexto de inteligencia artificial, varios estudios recientes demuestran que la ISO/IEC 25010 es fundamental para determinar la calidad de sistemas que integran Machine Learning, especialmente en ámbitos de gran importancia como la salud o la agricultura misma, donde una mala predicción puede causar daños irreparables [30].



Figura 8. *ISO/IEC 25010.*

Fuente: Elaboración propia.

En el trabajo actual la incorporación de la norma ISO/IEC 25010 será de gran importancia ya que se tiene la necesidad de validar la calidad de la aplicación web para la detección de enfermedades en cultivos de maíz, asegurando así que el software no solo tenga una buena precisión en la detección de enfermedades sino también cumpla con estándares de calidad como la eficiencia y la usabilidad, que permita un buen acogimiento por agricultores y técnicos expertos en el área.

2.6 Trabajos relacionados

Zhou (2024) [3] desarrollaron un modelo de inteligencia artificial (SNMPF) para la detección de enfermedades en cultivos de maíz, que combina una arquitectura basada en ShuffleNetV2 mejorada con Max Pooling y SimAM attention module. Realizaron estas mejoras y se integraron en la extracción de características de imágenes RGB que fueron tomadas a nivel de la hoja de la planta. El modelo alcanzó una precisión del 98,4% en la clasificación de las distintas enfermedades, superando en un 4,1 % al modelo original, demostrando que las arquitecturas livianas resultan ser bastante eficientes en el ámbito agrícola. Este estudio se limita al diagnóstico de imágenes tomadas desde el suelo, restringiendo de gran manera el alcance del diagnóstico en grandes extensiones de cultivos; además, no cuenta con la integración del modelo en una aplicación web para su uso en campo.

Askale (2025) [31] Desarrollaron una aplicación móvil en tiempo real para la clasificación de enfermedades en cultivos de maíz. Se evaluaron modelos como AlexNet, VGG16 y ResNet50. Cada modelo fue entrenado con un dataset que consta de 4188 imágenes. Redimensionaron las imágenes a 224 × 224 píxeles y utilizaron 50 épocas de entrenamiento bajo una tasa de aprendizaje de 0.000. El modelo que tuvo mejor desempeño fue VGG16, que alcanzó una precisión del 95 % y luego se desplegó en una aplicación Android. Esta investigación se limita a enfermedades que no existen en Ecuador. En modelos como ResNet50, que contienen una arquitectura más robusta, el dataset fue insuficiente; se podría implementar el uso de UAV para mayor inspección en campo real.

Weldeslasie (2025) [32] propone el desarrollo de un modelo híbrido para la detección de enfermedades en cultivos de maíz que mejora el diagnóstico mediante la combinación de una CNN y Vision Transformers. La metodología consistió en utilizar una CNN para la extracción de características locales de las imágenes y un ViT para capturar relaciones espaciales a escala global, empleando técnicas de preprocesamiento y de aumento de datos. El modelo tuvo muy buenas precisiones superando el 99 %, siendo ampliamente superior a arquitecturas simples como redes neuronales convolucionales tradicionales y Vision Transformers. Una de las limitaciones principales es que no cuenta con un entrenamiento para imágenes aéreas ni tampoco cuenta con una aplicación web o móvil, además de su alto costo computacional.

Gopalan (2025) [24] Propusieron desarrollar un modelo de inteligencia artificial para la detección de enfermedades en cultivos de maíz en el que agregaron una tecnología basada en IA explicable. Entrenaron una red convolucional ResNet152, en la que utilizaron imágenes RGB de hojas de maíz y aplicaron la técnica Grad-CAM para generar mapas de calor y mostrar las zonas en las que se fijó el modelo para dar un diagnóstico. El modelo alcanzó una precisión del 98.34 % durante las pruebas y el uso de Grad-CAM aumenta notablemente la confianza del diagnóstico que genera el modelo. Sin embargo, el modelo presenta una alta complejidad computacional, lo que limita su uso en entornos con recursos limitados, y no considera imágenes capturadas mediante UAV, además, carece de una validación en condiciones de campo reales.

Lu (2024) [26] Propusieron un nuevo método basado en imágenes RGB capturadas con UAV y una red neuronal para la detección de objetos para identificar plantas de maíz en diferentes etapas de crecimiento. Se utilizó un dron DJI para capturar imágenes del maíz; la altura de vuelo fue de 12 m sobre el nivel del suelo. Utilizaron el Segment Anything Model (SAM) para un etiquetado semiautomático y entrenaron un detector YOLOV5 con diferentes técnicas como el aumento de datos para mejorar la generalización del modelo. Las pruebas realizadas mostraron que el modelo alcanzó una precisión media (mAP@0.5 > 0.82) bajo condiciones reales de campo. Este trabajo se limita al conteo

de plantas, mas no a la detección de enfermedades; además, no cuenta con arquitecturas livianas y el dataset es muy limitado, con solo 200 imágenes por etapa.

Haque et al. (2022) [33] Propusieron un modelo de CNN ligero para la clasificación automática de los niveles de severidad de enfermedades presentes en las hojas de maíz; el objetivo fue ofrecer una herramienta capaz de funcionar en dispositivos con recursos limitados. La metodología consistió en crear un conjunto de datos específico de imágenes de hojas de maíz con diferentes grados de severidad e implementar una CNN eficiente con módulos modificados para extraer características discriminantes. Se comparó con modelos más pesados, como ResNet y DenseNet. El modelo alcanzó una precisión del 99.13 % en la clasificación de severidad. Su rendimiento fue superior al de modelos más pesados, pero con menor costo computacional. El modelo no incorpora técnicas de IA explicable; falta validación en campos reales, además, no aborda las enfermedades principales del Ecuador.

Theerthagiri (2024) [34] Entrenaron un modelo de IA para la clasificación de enfermedades en las hojas del maíz. Utilizaron un dataset con 4 clases (Roya, Tizon, Mancha gris y Sana) y, para el entrenamiento, aplicaron técnicas de aumento de datos. Se entrenaron varios modelos VGG16, ResNet34, ResNet50 y SqueezeNet. El modelo que tuvo la mayor precisión fue SqueezeNet, que alcanzó 97 % de precisión, superando entre 2 % y 5 % al resto de los modelos. Demostró que un modelo ligero y eficiente puede alcanzar muy buenas precisiones sin necesidad de recurrir a modelos pesados. El modelo entrenado no ha sido validado en condiciones de campo real; se podrían integrar arquitecturas híbridas, además de optar por el uso de imágenes UAV e integrar módulos de IA explicable.

CAPÍTULO III

3. MATERIALES Y MÉTODOS

En este capítulo se describe la metodología CRISP-DM (Cross-Industry Standard Process for Data Mining) utilizada para el desarrollo del proyecto, la cual está conformada por seis fases que van desde la comprensión del negocio hasta el despliegue de la solución, la primera fase parte del problema, los objetivos y el alcance definidos en el Capítulo I y los traduce en objetivos técnicos, mientras que las fases siguientes desarrollan la comprensión de los datos, su preparación, el modelado, la evaluación y, por último, el despliegue de la aplicación web para la detección de enfermedades en cultivos de maíz mediante arquitecturas híbridas. En la Tabla 3 se presentan las fases de la metodología CRISP-DM implementadas en el desarrollo del proyecto.

Tabla 3. *Fases de metodología CRISP-DM implementadas en el desarrollo del proyecto.*

Fase CRISP-DM	**Actividad Experimental Específica**	**Herramientas y Tecnologías**
Fase 1: Comprensión del Negocio	Definición del problema, los objetivos y el alcance del proyecto; evaluación de la situación y traducción de los objetivos del negocio en objetivos de minería de datos.	Revisión bibliográfica, Mendeley.
Fase 2: Comprensión de Datos	Recolección de imágenes mediante UAV en la comunidad de Pogllocunga, además del dataset público de Kaggle: "Corn Disease Drone Images". Análisis del dataset: número de imágenes, balance de clases y calidad del dataset.	Dron DJI Mini 3, Kaggle, Python
Fase 3: Preparación de Datos	Anotaciones de imágenes propias por clases, aumento de datos, división del dataset en train, val y test y resolución de imágenes (384 px)	Roboflow, Python, Google Colab, Google Drive, OpenCV.
Fase 4: Modelado	Diseño e implementación de una arquitectura híbrida MiT-B0 y DeepLabV3+, configuración de hiperparámetros, pesos para cada clase, Learning Rate, Transfer Learning, implementación de XAI y entrenamiento del modelo.	Python, PyTorch, Albumentations, Google Colab, Grad-CAM.
Fase 5: Evaluación	Evaluación cuantitativa del modelo a través de métricas (IoU) por clase. Generación de mapas de calor con Seg-Grad-CAM (XAI)	pytorch-grad-cam
Fase 6: Despliegue	Integración del modelo entrenado en la aplicación web. Desarrollo del frontend y backend. Pruebas funcionales del sistema completo.	React, FastAPI, PostgreSQL, Render, Supabase, Hugging Face Spaces.

Fuente: Elaboración propia.

3.1 Fase 1: Comprensión del Negocio

La primera fase de la metodología CRISP-DM corresponde a la comprensión del negocio, en la cual se busca entender el problema real que se quiere resolver antes de trabajar con los datos, de modo que el desarrollo del modelo responda a una necesidad concreta y no a un ejercicio aislado, para ello se parte del problema, los objetivos y el alcance definidos en el Capítulo I y se traducen en objetivos técnicos que guiarán las siguientes fases del proyecto.

3.1.1 Objetivos del negocio

El maíz representa una de las principales fuentes de sustento económico para las familias de la provincia de Imbabura; sin embargo, sus cultivos se ven afectados constantemente por enfermedades foliares como el tizón, la roya y la mancha blanca, que, cuando no se detectan a tiempo, reducen el rendimiento y la calidad de la cosecha. En la actualidad la mayoría de los agricultores diagnostican estas enfermedades de forma visual y empírica, lo que muchas veces lleva a identificar el problema demasiado tarde o de manera imprecisa y, como consecuencia, a un uso excesivo de agroquímicos que incrementa los costos de producción, por este motivo el objetivo del negocio es brindar una herramienta de apoyo que permita detectar de forma temprana, confiable y sencilla las enfermedades del maíz, con el fin de reducir las aplicaciones innecesarias de agroquímicos y proteger la productividad y la economía de los pequeños y medianos agricultores de la zona.

3.1.2 Evaluación de la situación

Antes de plantear la solución, se analizó la situación actual y los recursos disponibles para el desarrollo del proyecto, ya que en el campo el diagnóstico de las enfermedades depende de la experiencia del agricultor y no se cuenta con herramientas accesibles que apoyen esta tarea, mientras que las pocas soluciones existentes suelen requerir equipos costosos o conexión permanente a internet, lo que dificulta su uso en las zonas rurales de Imbabura. Para el desarrollo del modelo se dispuso de un dron DJI Mini 3 con cámara RGB para capturar imágenes aéreas de los cultivos en la comunidad de Pogllocunga, del dataset público Corn Disease Drone Images de Kaggle como complemento, y de herramientas gratuitas como Google Colab, Roboflow y PyTorch para el entrenamiento, lo que hizo viable el proyecto sin necesidad de una gran inversión en infraestructura. La primera jornada de recolección se realizó en un terreno de aproximadamente 2.5 hectáreas en la comunidad de Pogllocunga, mientras que la segunda jornada abarcó varias parcelas de menor tamaño en comunidades cercanas a Otavalo, ninguna de las cuales superó los 500 m². En cuanto al hardware, la captura de imágenes se realizó con un dron DJI Mini 3 equipado con una cámara RGB (sensor CMOS de 1/1.3 pulgadas y resolución de 12 MP), y el entrenamiento del modelo se ejecutó en la nube, en Google Colab Pro con una GPU NVIDIA A100, por lo que no fue necesario contar con un equipo local de altas prestaciones. En cuanto al software, para el modelo se utilizó Python con PyTorch, la librería Segmentation Models PyTorch (SMP), albumentations y pycocotools, además de Roboflow para la anotación de las imágenes; para la aplicación web se empleó React en el frontend, FastAPI en el backend y PostgreSQL como base de datos, desplegados en las plataformas Render, Supabase y Hugging Face Spaces.

3.1.3 Objetivos de minería de datos

Una vez comprendido el problema, el objetivo del negocio se tradujo en ~~un objetivo de minería de datos que consiste en~~ desarrollar un modelo de aprendizaje profundo basado en una arquitectura híbrida

que combina la extracción de características mediante redes convolucionales con el análisis global mediante mecanismos de atención tipo Transformer, según lo descrito en la sección 2.3.1, capaz de identificar, a nivel de píxel, las zonas afectadas por tizón, roya y mancha blanca, así como las zonas sanas, a partir de imágenes de maíz capturadas con dron, de esta forma el modelo debe segmentar y diferenciar cada enfermedad dentro de la imagen, indicando qué parte de la planta se encuentra afectada, para luego integrarse en una aplicación web que entregue el resultado de manera clara al usuario final.

3.1.4 Criterios de éxito del proyecto

Finalmente se establecieron los criterios que permitirían considerar el éxito del proyecto, formulados como indicadores verificables y agrupados en requisitos del modelo de inteligencia artificial y requisitos del sistema web. Para el modelo se definieron los siguientes umbrales mínimos: (i) un mIoU igual o superior a 0.70 sobre el conjunto de prueba, calculado con la métrica descrita en la sección 3.5.1; (ii) una matriz de confusión (tabla que compara las clases predichas frente a las reales) que evidencie un recall igual o superior al 60 % en cada una de las clases de enfermedad; y (iii) la generación de mapas de calor mediante Grad-CAM (visualización de las regiones de la imagen que más influyeron en la predicción del modelo) para al menos una imagen representativa por clase, que permitan verificar que las regiones de mayor activación coinciden con las zonas de lesión visible. Para el sistema web se definió que la aplicación debía alcanzar, en la evaluación con usuarios descrita en la sección 4.3, una valoración promedio igual o superior a 4.0 sobre 5 (nivel Alto según la escala de la sección 4.3.2) en las subcaracterísticas de usabilidad y eficiencia establecidas en la norma ISO/IEC 25010, así como una tasa de éxito en tareas igual o superior al 80 %, de modo que pueda ser utilizada por agricultores y técnicos sin conocimientos especializados. Todo el proceso se gestionó mediante las seis fases de la metodología CRISP-DM que se describen en este capítulo.

3.2 Fase 2: Comprensión de Datos

3.2.1 Recolección de imágenes propias mediante UAV

Previo a la recolección de imágenes, se realizó una búsqueda de diferentes cultivos de maíz alrededor de la provincia de Imbabura, se visitó varias comunidades cerca de la ciudad de Otavalo, otra visita fue a la Comunidad La Esperanza y por último se visitó varias comunidades de la vía Manzano Guarangui, en cada visita se realizó un análisis empírico del estado fitosanitario de los cultivos y se seleccionó únicamente los cultivos que presentaban síntomas de enfermedades útiles para este proyecto.

Primera Jornada de Recolección de Imágenes

La primera jornada se realizó el viernes 6 de marzo de 2026 en compañía del director del proyecto, Ing. Marco Remigio Pusdá Chulde, PhD. y el Sr. Jordán Alexis Manchimba, dueño del dron DJI Mini 3: el viaje fue a la vía Manzano-Guarangui, específicamente a la comunidad Pogllocunga, donde se encontraban dos cultivos pertenecientes al autor del proyecto. En dichos cultivos se identificó la presencia de la roya, que presenta síntomas de pequeñas pústulas de color óxido esparcidas sobre la hoja del maíz. Cabe mencionar que esta enfermedad no se encontraba distribuida en todo el cultivo, sino en zonas específicas, lo que condicionó la estrategia de vuelos con el dron. Las condiciones climáticas durante esta jornada fueron favorables, con variaciones leves entre momentos soleados y episodios de neblina ligera.



Figura 9. *Vuelo de dron DJI Mini 3.*

Fuente: Elaboración propia.

Los cultivos utilizados tenían una topografía irregular como se muestra en la *Figura 9*, por lo que no fue posible realizar vuelos automatizados en el dron y se optó por realizar vuelos manuales a alturas menores a 5 m sobre el cultivo como en el ejemplo *Figura 10*, esta decisión se dio por dos motivos: la primera fue por las condiciones irregulares del cultivo y la principal razón fue para lograr capturar los detalles de las enfermedades, ya que a más altura estos detalles eran imperceptibles, se realizó estos vuelos en un horario entre 9:00 a 12:00 para tener varios tipos de iluminación en las imágenes RGB.



Figura 10. *Imágenes capturadas por dron.*

Fuente: Elaboración propia.

Segunda Jornada de Recolección de Imágenes

La segunda jornada de recolección se realizó el miércoles 25 de marzo de 2026, de 9:00 a 13:00. En esta ocasión el viaje se realizó a dos comunidades cercanas a la ciudad de Otavalo, donde encontramos 5 cultivos con síntomas de la enfermedad Mancha Blanca, a diferencia de los cultivos de la primera jornada las plantas de maíz presentaron un desarrollo vegetativo significativamente superior como se puede observar en la *Figura 11*, por lo que se optó realizar vuelos manuales a una altura de 6 m con el dron DJI Mini 3, cuya cámara integrada cuenta con un sensor CMOS de 1/1.3 pulgadas y una resolución fotográfica de 12 MP, donde se logró recolectar un total de 211 Imágenes RGB. Las condiciones climáticas de esta jornada se mantuvieron soleadas de forma constante, aunque con una intensidad lumínica moderada. Las características de ambas jornadas de recolección se resumen en la Tabla 4.



Figura 11. *Toma de dron DJI Mini 3 en cultivos de Otavalo.*

Fuente: Elaboración propia.

Tabla 4. *Resumen de jornadas de recolección de imágenes.*

Jornada	Fecha	Ubicación	Altura de vuelo	Horario	Imágenes
1	06/03/2026	Pogllocunga	4 – 5 m	09:00 – 12:00	153
2	25/03/2026	Otavalo	6 m	09:00 – 13:00	211
Total de imágenes					364

Fuente: Elaboración propia.

3.2.2 Datasets públicos complementarios

En el capítulo I se investigaron los datasets públicos que podíamos utilizar para este proyecto. Estos conjuntos de datos se emplearon como complemento a las imágenes propias recolectadas mediante UAV, ya que estas últimas cubrieron únicamente las enfermedades de roya y mancha blanca observadas en campo, mientras que los datasets públicos permitieron incorporar la enfermedad del tizón del maíz,

ampliando la variedad de enfermedades representadas en el dataset final sin necesidad de nuevas jornadas de recolección. A diferencia de las imágenes propias, capturadas en condiciones y ubicaciones controladas por el equipo del proyecto (Pogllocunga y Otavalo), las imágenes públicas provienen de campos experimentales externos con condiciones de captura distintas, en los que encontramos la enfermedad del tizón del maíz, otra de las enfermedades que se fijó como objetivo; a continuación, se detalla cada una.

Dataset 01: Corn Disease Drone Images (Kaggle)

El primer dataset público seleccionado proviene de la plataforma Kaggle [7] ; este conjunto de datos cuenta con alrededor de 7670 imágenes RGB tomadas por un dron en un campo experimental de Nueva York, como se observa en la Figura 12. Inicialmente fue anotado para detección de objetos, sin embargo, para el proyecto no se utilizaron las etiquetas originales, sino solo las diferentes imágenes que se encontraban disponibles.

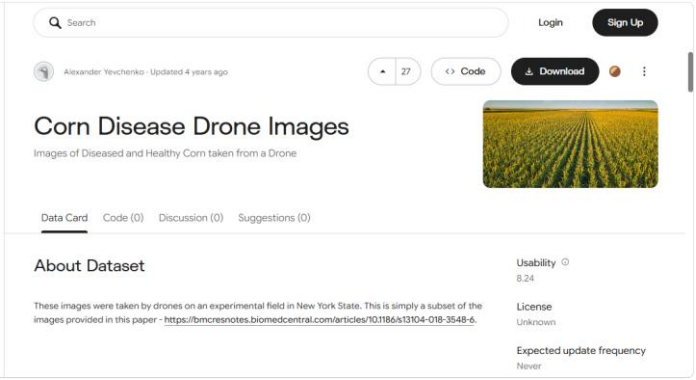


Figura 12. *Dataset público de Kaggle.*

Fuente: Corn Disease Drone Images.

Dado que el dataset es bastante grande, se optó por una selección manual de imágenes de mayor interés, como aquellas que presentaban síntomas de tizón, y se descartaron las que tenían fondos sin plantas de maíz, las que tenían hojas completamente sanas y las que tenían detalles muy difuminados. Este trabajo permitió seleccionar alrededor de 300 imágenes RGB representativas de la enfermedad buscada.

Dataset 02 Northern Leaf Blight UAV

Este dataset se logró encontrar en uno de los recursos que dejaron los autores del artículo científico "Quantitative Phenotyping of Northern Leaf Blight in UAV Images Using Deep Learning" [35] A diferencia del dataset anterior, este fue anotado para el uso en segmentación. Este conjunto de datos se

encontraba dividido en entrenamiento, con 2100 imágenes, y validación, con 400 imágenes, para un total de 2500 imágenes. A pesar de estar orientado a la segmentación, tampoco se usó el etiquetado original de los autores; solo se tomaron las imágenes como fuente de datos. Las características de los dos datasets públicos utilizados se resumen en la Tabla 5.

Tabla 5. *Resumen de imágenes recolectadas de datasets públicos.*

Dataset	Fuente	Total de imágenes	Imágenes utilizadas	Tipo de tarea original
Corn Disease Drone Images	Kaggle	7670	300	Detección de objetos
Northern Leaf Blight UAV	Artículo científico	2500	2500	Segmentación

Fuente: Elaboración propia.

3.2.3 Análisis de datos recolectados

Una vez reunidas las imágenes, tanto propias como de los diferentes datasets disponibles, se superaron las 3000 imágenes, en las que predominó la enfermedad del tizón, superando a las que presentaban enfermedades como la roya y la mancha blanca. En la Tabla 6 se describe la cantidad de imágenes y las fuentes de las que se obtuvo cada una.

Tabla 6. *Distribución total del dataset consolidado por clase y fuente.*

Clase	Imágenes propias	Kaggle	Artículo científico	Total
Roya	153	0	0	153
Mancha Blanca	211	0	0	211
Tizón foliar	0	300	2500	2800
Total	364	300	2500	3164

Fuente: Elaboración propia.

Como se puede observar en la *Tabla 6*, las enfermedades que se han recolectado fueron únicamente la roya y la mancha blanca, y la enfermedad del tizón se obtuvo únicamente de los datasets públicos, por lo que no fue necesario recolectarlas, debido a que hubo trabajos anteriores que abordaron esa enfermedad, facilitando la recolección de imágenes más rápidamente.

Para dar respuesta a lo solicitado respecto a la documentación de cada fuente de datos, la Tabla 7 consolida el origen, el número de imágenes utilizadas, la enfermedad representada, las características de captura y las restricciones de uso de las tres fuentes empleadas en el proyecto.

Tabla 7. *Resumen de origen, características de captura y restricciones de uso por fuente de datos.*

Fuente	Imágenes utilizadas	Enfermedad representada	Características de captura	Restricciones de uso
Imágenes propias (UAV)	364	Roya (153), Mancha blanca (211)	Dron DJI Mini 3 (sensor CMOS 1/1.3", 12 MP); vuelos manuales entre 4 y 6 m de altura; marzo de 2026; comunidades de Pogllocunga y Otavalo (Imbabura)	Datos propios del proyecto; disponibles para fines académicos citando esta tesis como fuente
Corn Disease Drone Images (Kaggle) [7]	300 (de 7670 disponibles)	Tizón foliar	Imágenes RGB capturadas por dron en el campo experimental Musgrave Research Farm (Universidad de Cornell, Nueva York); anotadas originalmente para detección de objetos, etiquetas no utilizadas	Dataset de acceso público en la plataforma Kaggle; uso permitido con fines de investigación citando la fuente original [7]
Northern Leaf Blight UAV (artículo científico) [35]	2500, luego descartada la mayoría en anotación de imágenes.	Tizón foliar	Imágenes UAV publicadas como material complementario del artículo "Quantitative Phenotyping of Northern Leaf Blight in UAV Images Using Deep Learning" [35]; divididas originalmente en entrenamiento (2100) y validación (400) para segmentación, etiquetas no utilizadas	Datos de acceso abierto asociados al artículo científico; uso permitido con fines de investigación citando la fuente original [35]

Fuente: Elaboración propia.

3.3 Fase 3: Preparación de Datos

3.3.1 Estrategia de preparación de imágenes por fuente de recolección

Dado que las imágenes fueron recolectadas de distintas fuentes y con distintas resoluciones, se optó por preparar cada una de forma individual, aplicando técnicas que nos permitieron obtener el máximo provecho de cada una. Esto fue necesario porque cada fuente presentaba condiciones de origen distintas: las imágenes propias capturadas con el UAV mostraban las zonas afectadas por la enfermedad ocupando

solo una parte de la fotografía de campo, mientras que las imágenes públicas de Kaggle tenían un tamaño excesivo para procesarlas directamente y las del artículo científico ya se encontraban preparadas por sus autores [35]; por ello, un tratamiento uniforme no habría aprovechado adecuadamente ninguna de las fuentes. Además, se descartaron las imágenes en las que el dron tomaba fotos muy alejadas de los cultivos o de baja calidad, en las que no es posible ver enfermedades presentes en las hojas del maíz.

Imágenes propias, recorte por zona de interés

Las imágenes recolectadas en campos reales con el dron DJI Mini 3 presentaban características especiales, donde las hojas del maíz con las enfermedades buscadas no se encontraban en toda la imagen por lo que se optó por realizar recortes por zonas de interés como se muestra en la Figura 13, todo este proceso se hizo manualmente, primero se identificó las zonas donde se concentraba la enfermedad deseada, para luego seleccionar recortar con herramientas tradicionales de Windows, sin embargo esta técnica tuvo un inconveniente, las imágenes que se iban guardando eran de dimensiones diferentes por lo que se utilizó Google Colab Pro para redimensionar todas esas imágenes a un tamaño de 512 × 512 pixeles cada una.



Figura 13. *Recorte de imágenes por zona de interés.*

Fuente: Elaboración propia.

Dataset de Kaggle: división en grids 2 x 2 Para el dataset de Kaggle se optó por otra técnica, luego de haber seleccionado las imágenes de mayor interés del dataset se revisó que las imágenes al ser tomadas por drones eran demasiado grandes para ser procesadas directamente para corregir esto se utilizó Google Colab Pro para dividir cada una de las imágenes en Grids de 2 x 2 como se muestra en la Figura 14, dando como resultado 4 subimágenes más pequeñas de 512 × 512 pixeles, este proceso logró aumentar el número de imágenes alrededor de 1000 siendo más eficientes para poderlas trabajar más adelante.



Figura 14. *Recorte de imágenes por grids.*

Fuente: Elaboración propia.

Dataset de artículo científico: uso directo

Las imágenes provenientes del artículo científico "Quantitative Phenotyping of Northern Leaf Blight in UAV Images Using Deep Learning" fueron utilizadas directamente ya que el trabajo realizado estaba orientado a segmentación por lo que los autores prepararon cada una de sus imágenes, teniendo un tamaño bastante eficiente para utilizar en el presente proyecto, el dataset también contaba con sus respectivas anotaciones, sin embargo, para este proyecto se decidió utilizar únicamente las imágenes originales y realizar anotaciones personales de cada una para mantener un trabajo único, sin embargo se usó un porcentaje muy bajo del total del dataset tomando únicamente las imágenes de validación y descartando por completo a las de entrenamiento, se usó estas imágenes únicamente para tener distintas variaciones en campos reales tomadas con UAV y tener un dataset propio más robusto.

3.3.2 Anotación de imágenes con Roboflow

El proceso de anotaciones de las imágenes se realizó a través de la plataforma web Roboflow [36]. Una herramienta que permite realizar distintos tipos de anotaciones, entre ellos la detección de objetos, la segmentación semántica y la segmentación de instancias, que es el módulo en el que se enfoca este proyecto, en el que cada enfermedad fue anotada manualmente, una por una, para obtener buenos resultados en el modelo de Deep Learning. El protocolo seguido en la plataforma consistió en: crear un proyecto de segmentación de instancias, cargar las imágenes recolectadas (propias y de datasets públicos), anotar manualmente cada lesión visible mediante polígonos asignándola a su clase correspondiente, revisar las anotaciones generadas y, finalmente, exportar el conjunto etiquetado en formato COCO (Common Objects in Context), un estándar que almacena en un único archivo JSON las imágenes, las categorías y las máscaras de segmentación por instancia, compatible con la librería pycocotools utilizada en el entrenamiento del modelo.

Para empezar a realizar anotaciones, se creó un proyecto con 3 clases diferentes, basadas en las enfermedades objetivas. En este paso, muchas de las imágenes no fueron anotadas, muchas de ellas porque no se podían observar rastros de enfermedad y muchas otras por no tener plantas de maíz y solo mostrar tierra. Para cada enfermedad se utilizó un enfoque diferente, siguiendo el lineamiento de segmentación de instancias. A continuación se explican las estrategias utilizadas para cada enfermedad.

Tizón: El tizón presenta formas irregulares a lo largo de la hoja, en forma de navaja, de color marrón oscuro, por lo que, para esta enfermedad, se delimitó mediante un polígono que la encerró por completo [37] Debido a que son imágenes de UAV, se podían observar una o varias hojas al mismo tiempo, por lo que el número de lesiones también aumentaba, por lo que en este enfoque se realizaba una o varias anotaciones para cubrir todas las lesiones presentes. En la Figura 15 se muestra un ejemplo del etiquetado de esta enfermedad en Roboflow.



Figura 15. *Etiquetado de tizón en Roboflow.*

Fuente: Elaboración propia.

Roya y Mancha Blanca: Para este tipo de enfermedades que se presentan en formas de pústulas pequeñas o manchas blancas en distintas zonas de la hoja sería un trabajo muy exhaustivo anotar individualmente cada una de ellas, por lo que se optó por encerrar en regiones las partes afectadas en un polígono, también hay casos especiales donde estas enfermedades están presentes en toda la hoja, para estos casos se delimitó toda la hoja afectada por un polígono grande, cada uno de los polígonos que se han realizado fueron anotados en su clase respectiva dependiendo de la enfermedad. En la Figura 16 se resume la estrategia de anotación empleada para cada enfermedad.

Roya (b) Mancha Blanca



Figura 16. *Estrategia de anotación para cada enfermedad.*

Fuente: Elaboración propia.

3.3.3 Preprocesamiento, división del dataset y aumento de datos

Finalmente, de las 3164 imágenes candidatas se descartaron aquellas con baja resolución, sin plantas de maíz visibles en el encuadre o sin rastros claros de enfermedad, quedando un conjunto anotado de 1256 imágenes. La clase Mancha Blanca presentó la mayor tasa de descarte (de 211 candidatas a 111 anotadas), debido a que una proporción considerable de sus imágenes originales presentaba baja resolución, encuadres sin tejido foliar visible o ausencia de síntomas identificables. Este conjunto se exportó en formato COCO, se redimensionó a 384 × 384 píxeles y se dividió en 80 % para entrenamiento, 10 % para validación y 10 % para pruebas, con el objetivo de tener un dataset que nos permita realizar pruebas con imágenes que no se toman en cuenta mientras entrenamos el modelo de IA. La Tabla 8 muestra la distribución de imágenes etiquetadas por clase en Roboflow.

Tabla 8. *Imágenes anotadas por clase en Roboflow.*

Clase	Imágenes anotadas
Roya	148
Mancha blanca	111
Tizón	1000
Total de etiquetas	1259
Total de imágenes únicas	1256

Fuente: Elaboración propia. Nota: la suma por clase (1259) supera el total de imágenes únicas (1256) porque 3 imágenes contienen simultáneamente lesiones de tizón y de roya en el mismo cuadro, por lo que se anotaron con ambas etiquetas.

Fuente: Elaboración propia.

Entre los varios ajustes que se hicieron en Roboflow está el redimensionamiento debido a que las imágenes que entraron a Roboflow están en varias escalas unas de 224 × 224 píxeles y otras de 512 × 512 píxeles, para corregir estas diferencias de escala se optó por redimensionar a 384 × 384 píxeles esto

debido a ser un término medio de la dimensión de las imágenes originales, además que con una redimensión media podríamos volver a redimensionar a escalas más grandes o más bajas al momento de entrenar esto gracias a librerías como albumentations, la redimensión de las imágenes dependerá de la arquitectura de entrenamiento que se haya armado, en el caso del proyecto actual se optó utilizar una dimensión de 384 × 384 píxeles para que la arquitectura definida pueda trabajar correctamente.

Además, se usó una de las opciones de Roboflow que permitió aumentar los datos para generar un dataset más grande. Entre las opciones que ofrecía Roboflow, se utilizaron estratégicamente aquellas que no perjudicaban la creación de un dataset robusto, teniendo en cuenta que las imágenes se toman en campos reales mediante UAV. Entre las opciones que se usaron se encuentran: Flip, Rotaciones, zoom, entre otras. Estos aumentos se seleccionaron cuidadosamente con el objetivo de simular diferentes condiciones de vuelo, como se observa en la *Tabla 9.*

Tabla 9. *Opciones de data augmentation seleccionadas en Roboflow.*

Transformación	Configuración	Justificación
Flip	Horizontal y Vertical	Simula distintas orientaciones de vuelo del dron
Rotación 90°	Horario y Antihorario	Cubre todas las orientaciones posibles de la cámara
Rotación libre	Entre -15° y +15°	Simula la inclinación natural en vuelos manuales
Recorte con zoom	0 % mínimo – 20 % máximo	Simula variaciones de altura durante el vuelo
Exposición	Entre -10 % y +10 %	Simula diferencias de iluminación entre jornadas

Fuente: Elaboración propia.

Cabe recalcar que estos aumentos de datos solo se hicieron en el conjunto de entrenamiento, evitando que las clases de validación y test tengan imágenes simuladas. Este aumento de datos se hizo a través de la plataforma Roboflow, aumentando el conjunto de entrenamiento en x3. En la *Tabla 10* se muestra la distribución de datos después de realizar un aumento de datos.

La secuencia de este trabajo evita filtraciones de datos entre los tres subconjuntos, ya que la partición 80/10/10 se realizó sobre las 1256 imágenes originales antes de aplicar cualquier aumento, de modo que cada imagen base pertenece exclusivamente a un subconjunto; el aumento se ejecutó después y únicamente sobre las imágenes ya asignadas al conjunto de entrenamiento, por lo que ninguna variante aumentada de una imagen de validación o de prueba pudo filtrarse hacia el entrenamiento, ni viceversa. En cuanto a las máscaras de segmentación, estas se generan a partir de los polígonos anotados en Roboflow y se almacenan en el archivo COCO como coordenadas de segmentación por instancia, que durante la carga del dataset (sección 3.4.3) se convierten en máscaras binarias por clase

a la resolución de entrada del modelo (384 × 384 píxeles), empleadas directamente para calcular la función de pérdida descrita en la sección 3.4.5.

Tabla 10. *Distribución de imágenes por subconjunto tras el aumento de datos en Roboflow.*

Subconjunto	Imágenes	Porcentaje
Entrenamiento aumento x3	3015	92.3 %
Validación	126	3.9 %
Prueba	125	3.8 %
Total	3266	100 %

Fuente: Elaboración propia.

3.4 Fase 4: Modelado y arquitectura de la Red

Se diseñó una arquitectura general con varios componentes reutilizables en otras arquitecturas. Se realizaron arquitecturas con varios modelos de Deep Learning para compararlos; los modelos seleccionados fueron CNN, Transformers y combinaciones de ambos.

Una vez concluida la preparación del dataset en la fase anterior se procedió con la construcción del modelo de Deep Learning encargado de realizar la segmentación de las enfermedades del maíz, para esta fase se utilizó como entorno principal Google Colab Pro aprovechando la disponibilidad de GPU para acelerar el proceso de entrenamiento, junto con la librería PyTorch como motor principal y la librería Segmentation Models PyTorch (SMP) que permite acceder a una variedad amplia de arquitecturas de segmentación preentrenadas. Todo el flujo del código se desarrolló de forma modular, lo que permitió probar diferentes combinaciones de encoders y decoders sin necesidad de modificar todo el proceso de entrenamiento, facilitando así la experimentación con distintos modelos antes de seleccionar la arquitectura final.

3.4.1 Selección y justificación de la arquitectura híbrida

Para la selección de la arquitectura se tuvieron en cuenta dos criterios fundamentales, el primero fue el rendimiento en tareas de segmentación semántica con imágenes de UAV y el segundo fue mantener una arquitectura liviana que pudiera ejecutarse sin problemas en dispositivos de bajos recursos, manteniendo así el enfoque principal del proyecto. Para ello, se experimentó con varias combinaciones de encoders convolucionales, tanto arquitecturas ligeras de uso extendido en la literatura (MobileNetV2, MobileOne-S0, ResNet18 y EfficientNet) como encoders basados en Transformers de la familia MiT (Mix Transformer), en sus variantes B0, B1 y B2. Finalmente, la combinación que mejor se ajustó a las necesidades del proyecto fue el uso del decoder DeepLabV3+ junto al encoder MiT-B0 (Mix Transformer B0), ambos disponibles en la librería Segmentation Models PyTorch.

Esta combinación representa un enfoque híbrido CNN-Transformers, donde el encoder MiT-B0 se encarga de extraer las características globales de la imagen mediante mecanismos de autoatención organizados en una estructura jerárquica de varias escalas, mientras que el decoder DeepLabV3+ refina las características obtenidas y reconstruye los mapas de segmentación a la resolución original. La principal ventaja de este enfoque es que el MiT-B0, a diferencia de los Vision Transformers tradicionales, está diseñado de manera jerárquica similar a una red convolucional, lo que permite capturar características a distintas escalas y al mismo tiempo mantener un costo computacional bajo. En total, el modelo cuenta con 4 136 820 parámetros entrenables, lo que es considerablemente menor que arquitecturas más pesadas como las basadas en ResNet101 o ViT-Base. Además, su tamaño aproximado es de 15.78 MB y su costo computacional es de 7.72 GFLOPs para una imagen de 384 × 384 píxeles, lo que lo hace adecuado para una posterior integración en una aplicación web sin requerir un alto consumo de recursos. Esta elección se valida cuantitativamente en la sección 4.2.2, donde se comparan la precisión de segmentación (mIoU), el número de parámetros, el costo computacional (GFLOPs), el tamaño del modelo y la velocidad de inferencia de esta arquitectura frente a otros seis codificadores (Tabla 53), confirmando que ofrece el mejor equilibrio entre desempeño y viabilidad de despliegue en la aplicación web.

3.4.2 Arquitectura del modelo DeepLabV3+ con encoder MiT-B0

El modelo construido sigue la estructura clásica encoder-decoder utilizada en tareas de segmentación. En la parte del encoder se utilizó el MiT-B0, que es la variante más liviana de la familia de modelos SegFormer, el cual procesa la imagen de entrada de 384 × 384 píxeles a través de cuatro etapas jerárquicas, donde cada etapa reduce la resolución espacial y aumenta la cantidad de canales, generando así mapas de características en distintas escalas, este enfoque jerárquico es muy similar al de una CNN tradicional pero con la ventaja de aplicar mecanismos de auto-atención eficientes, capaces de modelar relaciones globales sin necesitar las posiciones embebidas que se utilizan en los Transformers clásicos.

Por otro lado, el decoder DeepLabV3+ recibe los mapas de características generados por el encoder MiT-B0 y los procesa a través de un módulo ASPP (Atrous Spatial Pyramid Pooling), el cual aplica convoluciones con distintas dilataciones en paralelo para capturar el contexto multiescala, posteriormente las características de mayor resolución del encoder se combinan con las salidas del módulo ASPP para refinar los bordes de los mapas de segmentación, finalmente se aplica una convolución que reduce las características a las 4 clases definidas (Fondo, Tizón, Roya y Mancha Blanca) y se aplica una interpolación bilineal para devolver el mapa de segmentación a la resolución original de la imagen de entrada. En la Figura 17 se presenta el flujo general del modelo, desde la imagen de entrada hasta la máscara de segmentación.

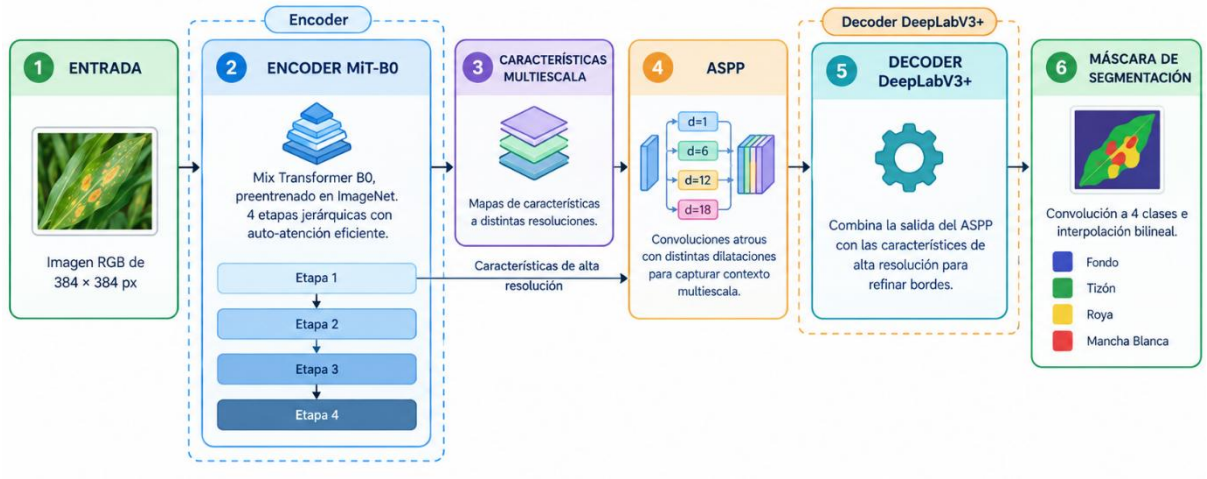


Figura 17. *Arquitectura del modelo DeepLabV3+ con encoder MiT-B0.*

Fuente: Elaboración propia.

El modelo se inicializó con pesos preentrenados en ImageNet para el encoder MiT-B0, aprovechando así la técnica de transfer learning, lo que permite acelerar la convergencia del entrenamiento y mejorar el rendimiento del modelo en datasets más pequeños como el que se utiliza en este proyecto. Esta técnica fue muy útil ya que el dataset propio no es tan extenso como los datasets utilizados originalmente para entrenar este tipo de modelos, y entrenar desde cero hubiese requerido una cantidad mucho mayor de imágenes y de poder de cómputo. En la Tabla 11 se presenta el resumen de la configuración general de la arquitectura utilizada.

Tabla 11. *Configuración general de la arquitectura.*

Componente	**Descripción**
Decoder	DeepLabV3+
Encoder	MiT-B0 (Mix Transformer B0)
Pesos preentrenados	ImageNet
Canales de entrada	3 (RGB)
Resolución de entrada	384 × 384 píxeles
Número de clases	4 (Fondo, Tizón, Roya, Mancha Blanca)
Parámetros totales	4 136 820
Tamaño aproximado	15.78 MB

Fuente: Elaboración propia.

3.4.3 Configuración del entorno y carga del dataset

Para empezar con el entrenamiento del modelo se preparó el entorno en Google Colab Pro, donde se aprovechó la disponibilidad de GPU A100 para acelerar el proceso, además se montó Google Drive

como almacenamiento principal, para poder cargar el dataset aumentado generado previamente en la fase anterior, posteriormente se copió la carpeta completa del dataset al almacenamiento local de Colab, esto se hizo para evitar tiempos lentos de lectura desde Drive, ya que cargar el dataset directamente desde Drive ralentizaba demasiado el entrenamiento.

Una vez cargado el dataset se construyó una clase personalizada en PyTorch versión 2.11.0 (MaizeDataset) que se encarga de leer cada una de las imágenes y sus respectivas anotaciones en formato COCO, esta clase utiliza la librería pycocotools para procesar las anotaciones y generar las máscaras de segmentación correspondientes, donde cada píxel se etiqueta con un valor entre 0 y 3 según la clase a la que pertenece, el valor 0 representa el fondo o las zonas sanas, el valor 1 corresponde al Tizón, el valor 2 a la Roya y el valor 3 a la Mancha Blanca. Posteriormente se construyeron los DataLoaders de entrenamiento y validación con un batch size de 10 imágenes. Este valor se eligió como un equilibrio entre el uso de memoria de la GPU en Google Colab Pro, la resolución de las imágenes y la estabilidad del entrenamiento. Para la reproducibilidad y trazabilidad del entrenamiento, se detallan a continuación los parámetros de configuración utilizados. El entrenamiento se ejecutó en Google Colab Pro con una GPU NVIDIA A100-SXM4 de 40 GB y cuDNN 9.27.0, usando Python 3.13.15 y PyTorch 2.11.0 (compilado para CUDA 13.0) junto a la librería Segmentation Models PyTorch (SMP) versión 0.5.0, además de albumentations 2.0.8, pycocotools 2.0.11, timm 1.0.30, NumPy 2.1.3 y OpenCV 5.0.0.93. No se fijó una semilla aleatoria (seed) explícita en el código de entrenamiento. Se entrenó durante un máximo de 70 épocas, con una resolución de entrada de 384 × 384 píxeles, seleccionada tras comparar cuatro resoluciones distintas (sección 4.2.3). Como optimizador se empleó AdamW, con tasas de aprendizaje diferenciadas entre encoder (6e-5) y decoder (6e-4), un calentamiento (warmup) lineal durante las primeras 3 épocas y un planificador de coseno para el resto del entrenamiento. La función de pérdida fue híbrida: durante las primeras 25 épocas se combinó Dice Loss y Cross-Entropy (0.5/0.5), y a partir de la época 26 se utilizó Lovász Loss, Cross-Entropy y Dice Loss (0.5/0.3/0.2). Se aplicó early stopping con una paciencia de 20 épocas sin mejora en el mIoU de validación; el entrenamiento se detuvo en la época 49 de las 70 planificadas, conservándose los pesos promediados mediante EMA (Exponential Moving Average) correspondientes a la época 29, de mejor desempeño en validación.

3.4.4 Aumento de datos en línea

Aunque el dataset ya contaba con un aumento previo realizado desde Roboflow, se decidió aplicar un aumento de datos adicional en línea durante el entrenamiento, este aumento se realizó utilizando la librería Albumentations [38] que permite aplicar transformaciones de forma aleatoria a cada imagen en cada época del entrenamiento, donde la idea principal de este enfoque es exponer al modelo a una mayor variedad de escenarios sin necesidad de almacenar las imágenes transformadas en disco; esto ayuda a reducir el riesgo de sobreajuste y mejora la capacidad de generalización del modelo en condiciones reales de campo.

Las transformaciones se aplicaron únicamente al subconjunto de entrenamiento, como se detalla en la *Tabla 12*, donde se observa que se aplicaron transformaciones geométricas y de iluminación. En cambio, para el subconjunto de validación únicamente se aplicaron la redimensión a 384 × 384 píxeles y la normalización; esto para evaluar el modelo siempre con imágenes en su estado original, sin transformaciones adicionales que pudieran alterar las métricas reales del modelo.

Tabla 12. *Aumento de datos en línea con Albumentations.*

Transformación	Probabilidad	Justificación
Resize	1.0	Estandariza el tamaño de entrada al modelo
HorizontalFlip	0.5	Simula vuelos del dron desde otra orientación
VerticalFlip	0.5	Simula vuelos del dron en sentido contrario
RandomRotate90	0.5	Cubre rotaciones de 90° en distintas direcciones
ShiftScaleRotate	0.5	Simula pequeños desplazamientos y rotaciones naturales del UAV
RandomBrightnessContrast	0.5	Simula variaciones de iluminación en distintas horas del día
Normalize	1.0	Normaliza los valores de píxeles según el preentrenamiento
ToTensorV2	1.0	Convierte la imagen a tensor para entrar al modelo

Fuente: Elaboración propia.

Cabe mencionar que la transformación de normalización fue muy importante, ya que el encoder MiT-B0 fue preentrenado con el dataset ImageNet, por lo que se deben normalizar las imágenes utilizando los mismos valores de media y desviación estándar que se usaron en ese preentrenamiento, en caso contrario, el modelo no podría aprovechar correctamente los pesos preentrenados y se afectaría su rendimiento.

3.4.5 Función de pérdida híbrida

Una de las decisiones más importantes a la hora de entrenar el modelo fue la elección de la función de pérdida; en este caso se optó por una pérdida híbrida por fases que combina Dice Loss, Lovász Loss y Cross Entropy Loss con pesos por clase, adaptando su composición a lo largo del entrenamiento. Durante las primeras 25 épocas, el modelo aprende la estructura general de las regiones mediante Dice y Cross Entropy; a partir de la época 26, la pérdida cambia de fase e incorpora Lovász Loss, que

optimiza directamente el índice de intersección sobre unión (IoU), es decir, la misma métrica que se busca maximizar (mIoU). Las fórmulas aplicadas en cada fase se muestran a continuación:

$$\mathcal{L}\,(\text{épocas } 1\text{–}25) = 0.5 \cdot \mathcal{L}\text{Dice} + 0.5 \cdot \mathcal{L}\text{CE}$$

$$\mathcal{L}\,(\text{épocas 26 en adelante}) = 0.5 \cdot \mathcal{L}\text{Lovász} + 0.3 \cdot \mathcal{L}\text{CE} + 0.2 \cdot \mathcal{L}\text{Dice}$$

Cada uno de los componentes cumple un rol fundamental durante el entrenamiento del modelo; a continuación, se explica el aporte de cada uno con mayor detalle.

- **Dice Loss:** Está presente en ambas fases (con un peso de 0.5 en la primera y de 0.2 en la segunda), ya que mide directamente la superposición entre la predicción del modelo y la máscara real; esto ayuda al modelo a generar máscaras más precisas y compactas, especialmente en clases con regiones pequeñas como la roya.
- Lovász Loss: Se incorpora en la segunda fase con un peso de 0.5; es un sustituto derivable del índice de Jaccard (IoU), por lo que optimiza de forma directa la métrica de evaluación principal del proyecto (mIoU). Al activarse una vez que el modelo ya ha aprendido la estructura general de las regiones, permite refinar los bordes y mejorar la superposición final de las máscaras predichas.
- Cross Entropy Loss con pesos por clase: Está presente en ambas fases (peso de 0.5 en la primera y de 0.3 en la segunda) e incorpora un suavizado de etiquetas (label smoothing) de 0.05 para evitar predicciones excesivamente confiadas. Se le aplicaron pesos personalizados por clase de [1.0, 2.0, 2.2, 2.8] para Fondo, Tizón, Roya y Mancha Blanca respectivamente, asignados de forma inversamente proporcional a la cantidad de píxeles que cada clase ocupa en el dataset. Estos pesos se suavizaron respecto a una configuración inicial más agresiva, ya que Dice y Lovász por sí solos ya compensan el desbalance de clases y unos pesos demasiado altos generaban falsos positivos en las clases minoritarias.

La combinación de estas tres funciones permite que el modelo se entrene de forma equilibrada, mejorando tanto la precisión global como el rendimiento en las clases minoritarias. Sin esta función de pérdida híbrida, el modelo tendía a predecir mayoritariamente la clase de fondo y a ignorar las enfermedades pequeñas como la roya, lo que reducía significativamente el rendimiento general del modelo.

3.4.6 Optimización y planificador de tasa de aprendizaje

Para optimizar los pesos del modelo durante el entrenamiento se utilizó el optimizador AdamW [39], que es una variante de Adam con una regularización L2 desacoplada (weight decay). Este

optimizador es muy utilizado en el entrenamiento de modelos basados en Transformers, ya que generalmente mejora la generalización del modelo. Se configuró con un weight decay de 1×10^{-2} (aplicado únicamente a los pesos de convolución y no a las normalizaciones ni a los sesgos, para no degradar las capas LayerNorm del encoder) y con tasas de aprendizaje diferenciadas: una tasa baja de 6×10^{-5} para el encoder preentrenado y una tasa mayor de 6×10^{-4} para el decoder inicializado de forma aleatoria, estrategia habitual al hacer fine-tuning de encoders basados en Transformers. La tasa de aprendizaje no se mantuvo en un valor fijo, ya que su ajuste se delegó al planificador descrito a continuación. Estos valores se eligieron tras realizar algunas pruebas previas para garantizar una buena convergencia del modelo sin caer en problemas de inestabilidad durante el entrenamiento.

Para el ajuste dinámico de la tasa de aprendizaje se utilizó un planificador secuencial compuesto por dos etapas: una fase de calentamiento lineal (warm-up) durante las primeras 3 épocas, en la que la tasa parte de un factor del 5 % de su valor base y sube progresivamente, seguida de un descenso mediante recocido por coseno (CosineAnnealingLR) hasta un valor mínimo de 1×10^{-6} al final del entrenamiento. El calentamiento evita saltos bruscos en las primeras épocas y el descenso por coseno permite afinar los pesos del modelo de manera más precisa. Este planificador se configuró para un total de 70 épocas y se actualiza en cada iteración (step) del entrenamiento. En la Tabla 13 se presenta la configuración completa del optimizador y del planificador.

Tabla 13. *Configuración del optimizador y planificador.*

Hiperparámetro	**Valor**
Optimizador	AdamW
Learning rate encoder	6×10^{-5}
Weight decay	1×10^{-2}
Planificador	Warm-up lineal + CosineAnnealingLR
Learning rate decoder	6×10^{-4}
Épocas máximas	70
Batch size	10

Fuente: Elaboración propia.

3.4.7 Implementación de IA Explicable (XAI) con Seg-Grad-CAM

Como se mencionó en el capítulo anterior, una de las grandes problemáticas en el uso de la inteligencia artificial en el sector agrícola es la falta de confianza por parte de los agricultores, ya que la mayoría de los modelos de Deep Learning funcionan como una caja negra donde el usuario solo ve el resultado pero no entiende cómo el modelo llegó a esa decisión, para resolver este problema se incorporó la técnica Grad-CAM (Gradient-weighted Class Activation Mapping) [24] adaptada al

modelo de segmentación semántica DeepLabV3+ con encoder MiT-B0, con el propósito de generar mapas de activación que permitan visualizar las regiones de la imagen asociadas con la predicción de una determinada enfermedad.

En la implementación desarrollada, Seg-Grad-CAM se aplica sobre las imágenes del conjunto de **prueba (test)** utilizando el modelo previamente entrenado y los pesos correspondientes al mejor modelo obtenido durante la validación. Para generar la explicación, primero se obtiene la máscara de segmentación predicha por el modelo mediante la clase con mayor probabilidad en cada píxel. A partir de esta predicción se selecciona una clase de enfermedad presente en la máscara (Tizón, Roya o Mancha Blanca) y se construye un objetivo mediante SemanticSegmentationTarget, utilizando la clase seleccionada y su máscara predicha. Posteriormente, se aplica Grad-CAM sobre la capa de salida de segmentación (segmentation_head[0]), donde se obtienen los gradientes asociados al objetivo definido. Estos gradientes se utilizan para determinar la importancia de las activaciones de la capa seleccionada y generar un mapa de activación espacial.

Finalmente, el mapa obtenido se normaliza y se superpone sobre la imagen original mediante show_cam_on_image, generando una representación visual en forma de mapa de calor. De esta manera, la explicación XAI permite identificar las regiones de la hoja que presentan mayor activación para la clase de enfermedad seleccionada y facilita la interpretación visual de la predicción realizada por el modelo. Para la evaluación visual se seleccionaron aleatoriamente cinco imágenes del conjunto de prueba que contienen al menos una de las enfermedades analizadas, mostrando para cada caso la imagen original, la etiqueta real, la predicción del modelo y el mapa generado mediante Seg-Grad-CAM. El esquema de funcionamiento de Seg-Grad-CAM para segmentación semántica se muestra en la Figura 18.

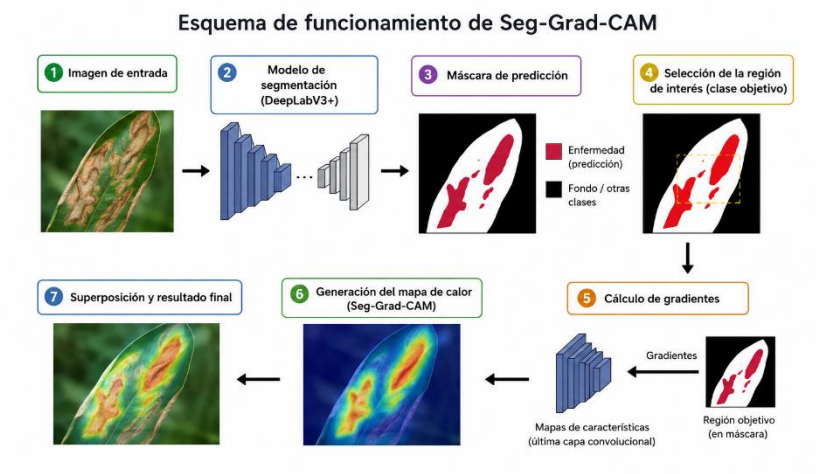


Figura 18. *Esquema de funcionamiento de Seg-Grad-CAM para segmentación semántica.*

Fuente: Elaboración propia.

El flujo de trabajo del módulo XAI se realiza de la siguiente manera: primero se realiza la predicción del modelo sobre la imagen de entrada y se obtiene el mapa de segmentación, posteriormente se identifica la clase predicha de mayor interés (que no sea fondo) y se genera el target de segmentación para esa clase, luego Grad-CAM utiliza los gradientes que fluyen hacia la última capa convolucional del decoder para calcular un mapa de calor que indica las regiones más importantes para la predicción, finalmente el mapa de calor se superpone a la imagen original generando una visualización clara de los puntos donde el modelo se centró para realizar el diagnóstico, esta integración fue muy útil ya que no solo se entrega un resultado final, sino también se entrega una evidencia visual del razonamiento del modelo.

3.4.8 Entrenamiento del modelo

Una vez que se contaba con la arquitectura, la función de pérdida, el optimizador, el planificador, las métricas y las estrategias de regularización configuradas, se procedió con el entrenamiento del modelo *(Figura 19)*, este se realizó en Google Colab Pro utilizando una GPU NVIDIA durante un máximo de 70 épocas, sin embargo, gracias a la estrategia de Early Stopping el entrenamiento se detuvo en la época 49, ya que el modelo no presentó mejoras significativas durante 20 épocas consecutivas, indicando que el modelo ya había alcanzado un buen nivel de rendimiento y continuar el entrenamiento podría llevar a un sobreajuste.

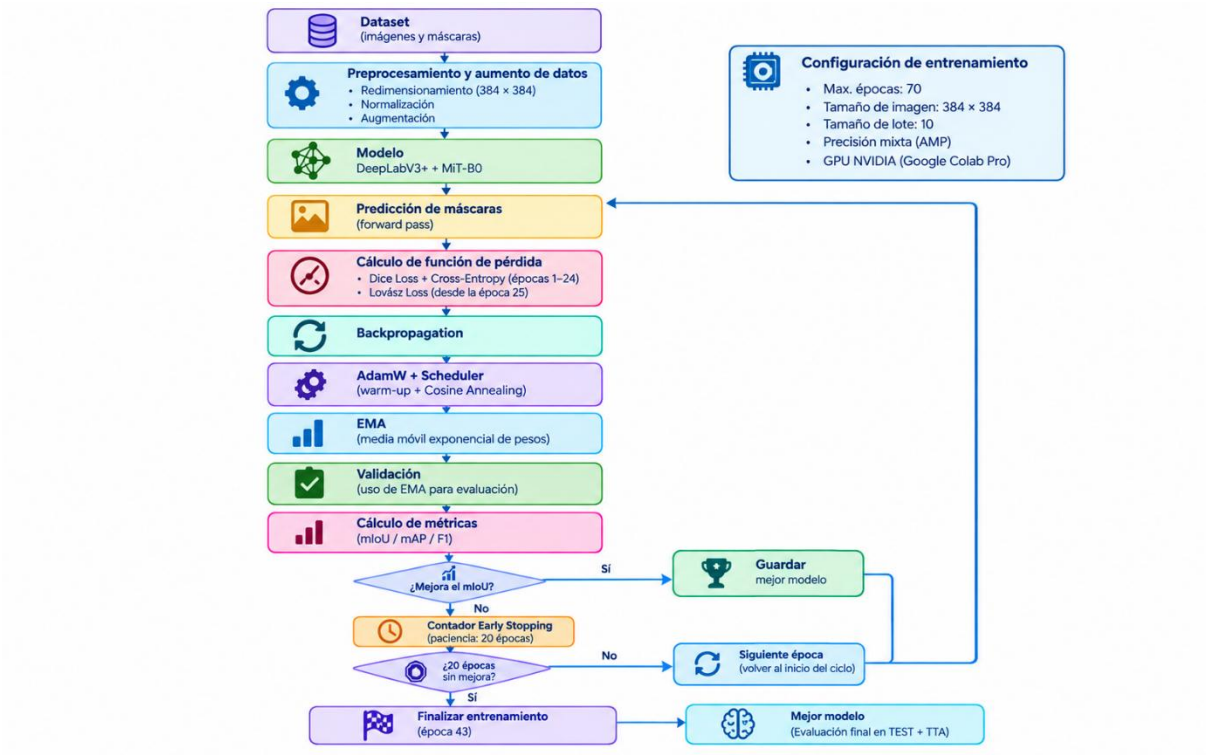


Figura 19. *Flujo del proceso de entrenamiento del modelo de segmentación semántica.*

Fuente: Elaboración propia.

3.5 Fase 5: Métricas de Evaluación

Una vez terminada la fase de modelado, se continuó con la quinta fase de la metodología CRISP-DM correspondiente a la evaluación, en la cual se buscó medir de forma objetiva el rendimiento del modelo entrenado para comprobar si los resultados obtenidos cumplían con los objetivos planteados al inicio de la investigación, para ello se analizó tanto el desempeño general del modelo como su comportamiento sobre cada una de las clases, apoyándose además en la matriz de confusión, en la comparación con las demás arquitecturas entrenadas y en la interpretación visual de las predicciones mediante los mapas de calor generados con técnicas de inteligencia artificial explicable. De esta forma se pudo conocer qué tan bien segmenta el modelo y qué tan confiables resultan sus predicciones al momento de detectar las enfermedades en imágenes reales de cultivos de maíz. Los resultados obtenidos en esta fase se presentan y analizan en detalle en el Capítulo IV.

3.5.1 Métricas de evaluación utilizadas

Para medir el rendimiento del modelo de segmentación se utilizaron varias métricas propias de las tareas de segmentación semántica, ya que cada una permite analizar un aspecto diferente del comportamiento del modelo, tanto a nivel general como en cada una de las clases consideradas: Fondo,

Tizón, Roya y Mancha Blanca. El uso combinado de estas métricas permitió obtener una visión más completa del desempeño y evitar que la evaluación dependiera de un único valor que pudiera ocultar errores en las clases con menor cantidad de muestras; de esta forma, a continuación, se describe cada una de las métricas utilizadas:

IoU (Intersección sobre Unión): Es una de las métricas más representativas en segmentación, ya que mide el grado de superposición entre la región predicha por el modelo y la región real de la máscara, calculando la relación entre el área de intersección y el área de unión de ambas, de modo que un valor cercano a 1 indica que la predicción coincide casi por completo con la zona real de la enfermedad.

mIoU (IoU medio): Corresponde al promedio del IoU de todas las clases, por lo que se utilizó como métrica principal para valorar el rendimiento global del modelo, ya que resume en un solo valor qué tan bien se realiza la segmentación en general.

Precisión: Indica la proporción de píxeles que el modelo clasificó correctamente dentro de una clase respecto al total de píxeles que predijo para esa clase; un valor alto significa que, cuando el modelo señala una zona como afectada, generalmente acierta.

Recall: mide la capacidad del modelo para detectar todos los píxeles que realmente pertenecen a una clase, por lo que un valor alto indica que el modelo no deja sin identificar zonas afectadas que sí se encuentran presentes en la hoja.

F1-score: Combina la precisión y el recall en un solo valor mediante su media armónica, lo que resulta especialmente útil cuando existe un desbalance entre las clases, ya que penaliza los casos en los que una de las dos métricas es baja.

Exactitud de píxeles (Pixel Accuracy): Representa la proporción de píxeles correctamente clasificados sobre el total de píxeles de la imagen; sin embargo, esta métrica se interpretó con cuidado, ya que al existir una clase de fondo predominante puede mostrarse muy alta, aunque las enfermedades no se hayan segmentado de forma adecuada.

AP y mAP (precisión promedio): La precisión promedio (AP) resume el comportamiento del modelo para una clase considerando distintos umbrales, mientras que el mAP corresponde al promedio de la AP de todas las clases, complementando de esta forma la información aportada por el mIoU.

3.5.2 Proceso de evaluación del modelo

La evaluación se realizó sobre el conjunto de validación que se había separado previamente durante la fase 3.3 de preparación de los datos, ya que estas imágenes no fueron utilizadas durante el ajuste de

los pesos del modelo y permiten medir su capacidad real de generalización frente a datos nuevos. De esta forma se evitó evaluar el modelo con las mismas imágenes con las que fue entrenado, lo que habría dado resultados poco confiables y demasiado optimistas. Durante el entrenamiento se monitoreó el comportamiento del modelo en cada época y se conservaron los pesos correspondientes al mejor desempeño obtenido mediante la estrategia de Early Stopping, de modo que sobre ese modelo final se aplicaron las métricas descritas anteriormente para obtener los resultados definitivos. Además, se utilizó la matriz de confusión como herramienta complementaria, ya que permite observar de forma más detallada en qué clases el modelo presenta mayores confusiones al momento de asignar cada píxel.

Para el cálculo final de las métricas se utilizaron los pesos promediados mediante EMA (Exponential Moving Average), correspondientes a la época con mejor desempeño en validación, en lugar de los pesos de la última época de entrenamiento. Adicionalmente, durante la inferencia se aplicó aumento en el tiempo de prueba (TTA, Test-Time Augmentation), promediando las predicciones obtenidas sobre las ocho vistas generadas por las transformaciones del grupo diédrico D4 (rotaciones de 90°, 180° y 270°, y sus reflexiones), lo que permite obtener predicciones más estables y consistentes.

3.5.3 Evaluación comparativa de las arquitecturas

Además de evaluar el modelo final, se realizó una comparación entre las distintas arquitecturas entrenadas con el fin de justificar la elección del encoder utilizado; para ello se entrenaron bajo las mismas condiciones varias configuraciones diferentes, modificando únicamente el encoder, de modo que se pudieran comparar de forma justa los encoders MiT-B0 y MiT-B1, pertenecientes a la familia de transformers jerárquicos, junto con el encoder convolucional EfficientNet-B0 y otras arquitecturas similares con tamaño de parámetros similares teniendo en cuenta que deben ser arquitecturas livianas. Esta comparación permitió analizar no solo la precisión alcanzada por cada arquitectura, sino también su costo computacional, considerando la cantidad de parámetros y el tamaño final del modelo, ya que uno de los objetivos del proyecto era mantener un modelo liviano que pudiera integrarse en la aplicación web sin generar un alto consumo de recursos. Los resultados obtenidos en esta comparación se presentan y analizan en detalle en el Capítulo IV.

3.5.4 Evaluación cualitativa mediante mapas de calor

De manera complementaria a las métricas cuantitativas, se realizó una evaluación cualitativa de las predicciones del modelo mediante los mapas de calor generados con la técnica Seg-Grad-CAM, cuyo protocolo se detalló en la sección 3.4.7, ya que estos permiten verificar visualmente si el modelo está enfocando su atención en las zonas correctas de la hoja al realizar una predicción. Esta evaluación se consideró importante porque un modelo puede alcanzar métricas elevadas y, aun así, tomar decisiones basadas en elementos del fondo o en características que no corresponden a los síntomas reales de la

enfermedad; por lo tanto, el análisis visual se empleó como criterio adicional para verificar si las zonas de mayor activación coincidían con las regiones afectadas del follaje. Los hallazgos de este análisis se presentan y discuten en la sección 4.2.7.

3.6 Fase 6: Despliegue

Una vez que el modelo fue evaluado y se comprobó que cumplía con los objetivos planteados, se pasó a la sexta y última fase de la metodología CRISP-DM, que corresponde al despliegue, en la cual el modelo se llevó a producción dentro de una aplicación web funcional para que pueda ser utilizada por los usuarios finales y no se quede únicamente como un experimento, de esta forma la aplicación permite que cualquier persona suba una imagen de su cultivo y obtenga el resultado del análisis de manera rápida y sencilla.

Para el despliegue de la aplicación se construyó una arquitectura cliente-servidor conformada por tres partes, un frontend en el que el usuario interactúa con el sistema, un backend que se encarga de cargar el modelo MiT-B0 y de realizar la inferencia, y una base de datos en la que se almacenan los usuarios y el historial de los análisis, de modo que el frontend quedó desplegado en la plataforma Render, el backend en Hugging Face Spaces y la base de datos PostgreSQL alojada en Supabase, de esta forma cuando el usuario sube una imagen tomada con dron esta se envía al backend, en donde el modelo la procesa y devuelve la enfermedad detectada, el porcentaje de área afectada y el mapa de calor generado con Grad-CAM. En la Figura 20 se presenta el flujo de despliegue del sistema, que va desde el archivo de la imagen del cultivo hasta la devolución del resultado al usuario en la aplicación web, pasando por la API y el servicio de inferencia.

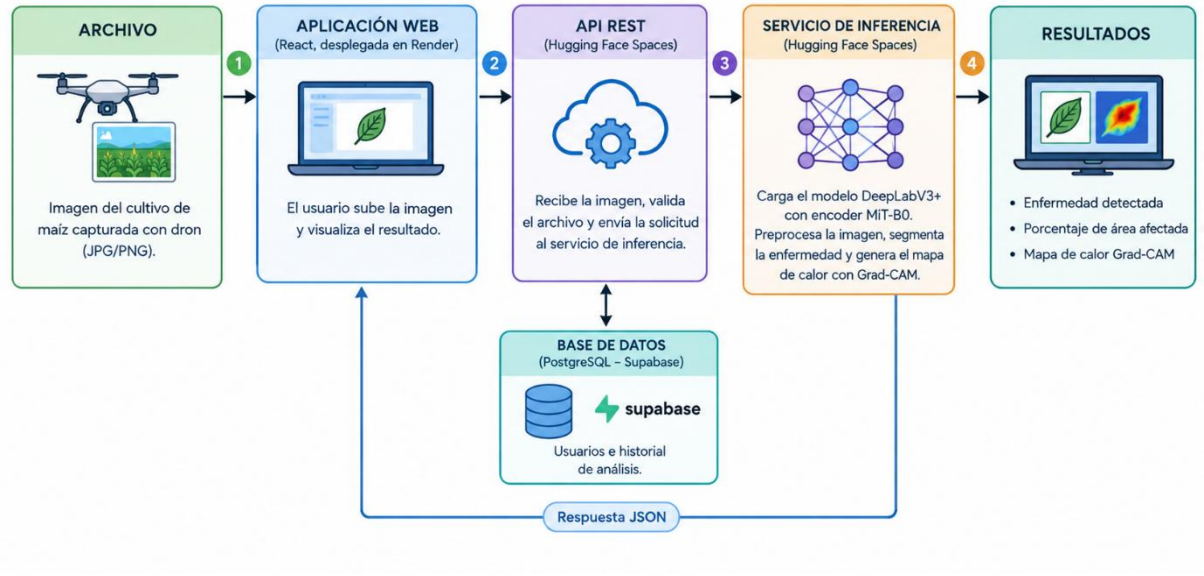


Figura 20. *Flujo de despliegue del sistema AgroScan.*

Fuente: Elaboración propia.

En este apartado se presenta de forma general cómo quedó desplegada la aplicación, con el fin de cerrar el ciclo de la metodología CRISP-DM.

3.7 Metodología Scrum para el desarrollo de la aplicación web

Si bien la metodología CRISP-DM, expuesta en los apartados anteriores, sirvió para estructurar el ciclo completo del modelo de inteligencia artificial, la creación de la aplicación web encargada de acercar ese modelo a los usuarios finales se apoyó en Scrum como marco ágil de desarrollo. Ambos enfoques se complementaron de forma natural dentro del proyecto, puesto que CRISP-DM se centra en el tratamiento de los datos y el entrenamiento del modelo, mientras que Scrum estructura la construcción del software en ciclos cortos e incrementales, lo que permitió planificar los requerimientos, jerarquizar las funcionalidades y revisar los avances a medida que se implementaban los módulos del sistema. En la Tabla 14 se muestra cómo se distribuyó cada metodología entre los componentes del proyecto.

Tabla 14. *Metodologías utilizadas para el proyecto.*

Componente	Aplicación metodológica
Modelo de inteligencia artificial	CRISP-DM para los datos, el entrenamiento y la evaluación del modelo.
Aplicación web	Scrum para organizar el desarrollo por iteraciones.
Integración del sistema	Trabajo combinado entre las dos metodologías.

Fuente: Elaboración propia.

Para construir la aplicación web, se siguieron las tres etapas que plantea Scrum: Pre-Juego, Juego y Post-Juego. Durante el Pre-Juego se fijaron los roles del equipo, los requerimientos, las historias de

usuario, el Product Backlog y una primera versión de la arquitectura; en la etapa de Juego se concentró la programación de los módulos a través de sprints, de modo que el avance se fue entregando por partes; por último, el Post-Juego reunió las pruebas funcionales, los ajustes finales, la puesta en producción del sistema y la preparación de la evaluación con usuarios conforme a la norma ISO/IEC 25010.

3.7.1 Definición del equipo Scrum

Antes de iniciar la construcción del sistema, se establecieron los roles principales del equipo Scrum, tomando en cuenta la manera en que cada participante intervendría en la planificación, la revisión y la implementación de la aplicación. Distribuir de esta forma las responsabilidades facilitó ordenar el trabajo del proyecto y llevar un control continuo de los avances logrados en cada iteración. En la Tabla 15 se detalla la conformación del equipo Scrum definido para este proyecto.

Tabla 15. *Equipo Scrum del proyecto.*

Rol	Integrante / actor	Responsabilidad
Product Owner	Ing. Marco Remigio Pusdá Chulde, PhD.	Supervisar la evolución del proyecto y validar los avances.
Scrum Master	Alex Ronaldo Luna Quilumba	Coordinar las actividades y dar seguimiento al desarrollo del sistema.
Equipo de desarrollo	Alex Ronaldo Luna Quilumba	Diseñar, codificar e integrar los distintos módulos de la aplicación.
Stakeholders	PhD. Iván Danilo García Santillán (asesor) Agricultores y técnicos agrícolas usuarios de la aplicación.	Brindar observaciones y retroalimentación sobre el funcionamiento del sistema.

Fuente: Elaboración propia.

Definición de nomenclatura

Como la planificación exigía identificar con claridad cada artefacto generado durante el desarrollo, se adoptó una nomenclatura propia para codificarlos. Gracias a esta codificación, resultó más sencillo conservar el orden y rastrear la relación existente entre los requerimientos, las historias de usuario y los sprints. En la Tabla 16 se detallan los códigos empleados para nombrar los artefactos de planificación.

Tabla 16. *Nomenclatura utilizada para la planificación Scrum.*

Código	Descripción
RF	Requerimiento funcional del sistema
RNF	Requerimiento no funcional del sistema
HU	Historia de usuario
SP	Sprint de desarrollo

Fuente: Elaboración propia.

3.7.2 Fase de Pre-Juego: planificación inicial

En esta etapa se prepararon los elementos que servirían de base para levantar la aplicación web, pues en ella se establecieron la nomenclatura, los requerimientos, las historias de usuario, el Product Backlog y la arquitectura del sistema, con el propósito de contar con una guía clara antes de comenzar el desarrollo por sprints.

Requerimientos funcionales

Los requerimientos funcionales establecen las acciones que el usuario está en capacidad de realizar dentro de la aplicación web. En este proyecto, dicho alcance se concretó en tareas específicas como la autenticación de usuarios, la carga de imágenes obtenidas con dron, el diagnóstico de enfermedades a través del modelo de inteligencia artificial, la visualización del mapa de calor, la revisión del historial de análisis y el acceso a información sobre las enfermedades del maíz. En la Tabla 17 se especifican las funciones principales que delimitan el alcance operativo del sistema.

Tabla 17. *Requerimientos funcionales.*

Código	Descripción
RF01	Habilitar el registro y el inicio de sesión de los usuarios como paso previo para acceder a las funciones del sistema.
RF02	Admitir la carga de imágenes de cultivos de maíz obtenidas con UAV para someterlas posteriormente al análisis.
RF03	Analizar la imagen cargada con el modelo de inteligencia artificial y presentar la enfermedad identificada junto con el porcentaje de área afectada.
RF04	Exhibir el mapa de calor producido con Grad-CAM para evidenciar las regiones en las que el modelo centró su atención al diagnosticar.
RF05	Registrar y poner a disposición de cada usuario el historial de los análisis efectuados.
RF06	Disponer de una sección informativa acerca de las enfermedades del maíz contempladas en el proyecto.
RF07	Ofrecer una página principal que, tras el inicio de sesión, presente accesos directos a las funciones del sistema y el estado general de la plataforma.
RF08	Permitir al usuario gestionar su cuenta, actualizando su nombre y cambiando su contraseña de forma segura.
RF09	Ofrecer la posibilidad de alternar entre un tema claro y uno oscuro en la interfaz de la aplicación.
RF10	Permitir el análisis integral de una parcela completa delimitada sobre un mapa satelital, generando puntos de muestreo, diagnosticando cada uno con el modelo de inteligencia artificial y consolidando la enfermedad dominante y la severidad global del lote.

Fuente: Elaboración propia.

Requerimientos no funcionales

Los requerimientos no funcionales fijan las condiciones de calidad que la aplicación web debe satisfacer más allá de sus funciones esenciales, y se relacionan con aspectos tales como la usabilidad, la seguridad, el rendimiento, la persistencia de los datos y la disponibilidad del sistema. En la Tabla 18 se describen estas condiciones de calidad definidas para el funcionamiento de la aplicación.

Tabla 18. *Requerimientos no funcionales.*

Código	Descripción
RNF01	La interfaz debe resultar intuitiva y sencilla de manejar tanto para agricultores como para técnicos agrícolas.
RNF02	El ingreso al sistema debe efectuarse a través de una autenticación segura basada en correo electrónico y contraseña.
RNF03	Tanto los datos de los usuarios como el historial de análisis deben conservarse de forma persistente en una base de datos PostgreSQL.
RNF04	El diagnóstico debe generarse con el modelo de inteligencia artificial alojado en el backend, entregando el resultado dentro de un tiempo razonable.
RNF05	La aplicación debe poder utilizarse desde un navegador web sin que se requiera instalación adicional.
RNF06	El sistema debe valorarse conforme a la norma ISO/IEC 25010 en sus características de usabilidad y eficiencia.

Fuente: Elaboración propia.

Historias de usuario

Las historias de usuario sirvieron para describir las funcionalidades de la aplicación desde el punto de vista de quien la utiliza. Cada una de ellas se redactó considerando el actor implicado, la necesidad detectada, la prioridad asignada, la estimación del esfuerzo y los criterios con los que se validaría su cumplimiento. En la Tabla 19 se presenta la estructura empleada para documentar cada historia de usuario.

Tabla 19. *Formato de historias de usuario.*

Campo	Descripción
Número	HUXX
Usuario	Actor que utiliza la funcionalidad
Nombre de la historia	Nombre breve de la funcionalidad
Prioridad en el negocio	Alta / Media / Baja
Riesgo en desarrollo	Alto / Medio / Bajo
Puntos estimados	Valor en Story Points (1, 2, 3, 5, 8, 13)
Iteración asignada	Sprint en el que se desarrolla la historia
Programador responsable	Persona encargada de implementar la historia
Descripción	Necesidad expresada desde la perspectiva del usuario
Validación	Criterios para comprobar el cumplimiento de la historia

Fuente: Elaboración propia.

Para estimar el esfuerzo asociado a cada historia de usuario se recurrió a los Puntos de Historia (Story Points), una unidad de medida relativa que valora la complejidad, el volumen de trabajo y la incertidumbre de cada funcionalidad en lugar de expresarla en horas exactas. Esta técnica resulta apropiada para el desarrollo ágil, ya que permite comparar las historias entre sí y estimar la capacidad del equipo en cada sprint sin depender de una duración fija, lo que reduce el margen de error propio de las estimaciones basadas en tiempo. Para asignar los valores se empleó una escala fundamentada en la secuencia de Fibonacci (1, 2, 3, 5, 8 y 13), en la que un mayor número de puntos refleja una funcionalidad más compleja o con mayor riesgo, tal como se detalla en la Tabla 20.

Tabla 20. *Escala de estimación en Story Points.*

Puntos de historia	**Descripción**
1 – 2	Complejidad baja: funcionalidad sencilla, con esfuerzo y riesgo mínimos.
3 – 5	Complejidad media: funcionalidad moderada, con esfuerzo intermedio.
8	Complejidad alta: funcionalidad compleja, con mayor esfuerzo e incertidumbre.
13	Complejidad muy alta: conviene dividir la historia en otras más pequeñas.

Fuente: Elaboración propia.

Seguidamente se exponen las historias de usuario elaboradas a partir de los requerimientos funcionales de la aplicación web, donde cada una expresa una necesidad concreta del usuario y se enlaza con una funcionalidad construida a lo largo de los sprints.

En la Tabla 21 se describe el acceso inicial del usuario a la aplicación.

Tabla 21. *Historia de usuario HU01.*

Número	**HU01**
Usuario	Usuario de la aplicación
Nombre de la historia	Registro e inicio de sesión
Prioridad en el negocio	Alta
Riesgo en desarrollo	Medio
Puntos estimados	5
Iteración asignada	1
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero crear una cuenta e iniciar sesión con mi correo electrónico y contraseña para ingresar de manera segura a las funciones del sistema.
Validación	• Comprobar que un nuevo usuario logre registrarse con su correo y contraseña. • Verificar el ingreso al sistema con las credenciales previamente creadas. • Constatar que los datos del usuario queden guardados en la base de datos PostgreSQL. • Corroborar que el acceso a las funciones protegidas se controle correctamente.

Fuente: Elaboración propia.

En la Tabla 22 se describe la carga de imágenes que da inicio al proceso de diagnóstico.

Tabla 22. *Historia de usuario HU02.*

Número	**HU02**
Usuario	Usuario de la aplicación
Nombre de la historia	Carga de imágenes del cultivo
Prioridad en el negocio	Alta
Riesgo en desarrollo	Bajo
Puntos estimados	3
Iteración asignada	2
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero subir una imagen de mi cultivo de maíz tomada con dron para emplearla en el proceso de diagnóstico de enfermedades.
Validación	• Constatar que el usuario pueda elegir una imagen desde su dispositivo. • Verificar que se valide el formato de la imagen cargada. • Corroborar que la imagen se pueda previsualizar antes de comenzar el análisis.

Fuente: Elaboración propia.

En la Tabla 23 se describe el análisis inteligente de las imágenes para detectar enfermedades.

Tabla 23. *Historia de usuario HU03.*

Número	**HU03**
Usuario	Usuario de la aplicación
Nombre de la historia	Diagnóstico con inteligencia artificial
Prioridad en el negocio	Alta
Riesgo en desarrollo	Alto
Puntos estimados	8
Iteración asignada	2
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero procesar la imagen cargada para reconocer las enfermedades presentes en las hojas de maíz mediante el modelo de inteligencia artificial incorporado al sistema.
Validación	• Comprobar que el modelo procese la imagen en el backend. • Verificar que se detecte la enfermedad y se calcule el área afectada. • Corroborar que el resultado se presente con claridad al usuario.

Fuente: Elaboración propia.

En la Tabla 24 se describe la visualización del mapa de calor asociado al diagnóstico.

Tabla 24. *Historia de usuario HU04.*

Número	**HU04**
Usuario	Usuario de la aplicación
Nombre de la historia	Visualización del mapa de calor
Prioridad en el negocio	Alta
Riesgo en desarrollo	Medio
Puntos estimados	5
Iteración asignada	3
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero observar el mapa de calor generado con Grad-CAM para conocer las zonas en las que el modelo se concentró durante el diagnóstico y así confiar en el resultado.
Validación	• Constatar que el mapa de calor se genere sobre la imagen analizada. • Verificar que las zonas resaltadas coincidan con las áreas afectadas. • Corroborar que el mapa de calor se muestre con claridad junto al resultado.

Fuente: Elaboración propia.

En la Tabla 25 se describe el seguimiento de los análisis efectuados.

Tabla 25. *Historia de usuario HU05.*

Número	**HU05**
Usuario	Usuario de la aplicación
Nombre de la historia	Historial de análisis
Prioridad en el negocio	Media
Riesgo en desarrollo	Bajo
Puntos estimados	3
Iteración asignada	3
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero revisar los análisis efectuados con anterioridad para consultar los resultados y dar seguimiento al estado de mis cultivos.
Validación	• Comprobar que los análisis queden almacenados y vinculados al usuario correspondiente. • Verificar que se muestre el listado de los análisis efectuados. • Corroborar que cada registro despliegue la información principal del resultado.

Fuente: Elaboración propia.

En la Tabla 26 se describe la sección informativa sobre las enfermedades del maíz.

Tabla 26. *Historia de usuario HU06.*

Número	**HU06**
Usuario	Usuario de la aplicación
Nombre de la historia	Información sobre enfermedades
Prioridad en el negocio	Baja
Riesgo en desarrollo	Bajo
Puntos estimados	2
Iteración asignada	4
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero disponer de información sobre las enfermedades del maíz para entender mejor los diagnósticos que entrega el sistema.
Validación	• Constatar que se pueda acceder a la sección informativa. • Verificar que en ella se describan las enfermedades contempladas en el proyecto. • Corroborar que la información resulte clara y comprensible.

Fuente: Elaboración propia.

En la Tabla 27 se describe el acceso a la página principal de la aplicación.

Tabla 27. *Historia de usuario HU07.*

Número	**HU07**
Usuario	Usuario de la aplicación
Nombre de la historia	Página principal
Prioridad en el negocio	Media
Riesgo en desarrollo	Bajo
Puntos estimados	3
Iteración asignada	5
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero contar con una página principal que me reciba tras iniciar sesión, donde pueda visualizar accesos directos a las funciones del sistema y el estado general de la plataforma.
Validación	• Comprobar que la página principal se muestre después de iniciar sesión. • Verificar que ofrezca accesos directos a la carga de imágenes y al historial. • Corroborar que presente el estado del sistema y la información de contexto del usuario.

Fuente: Elaboración propia.

En la Tabla 28 se describe la gestión de la cuenta del usuario.

Tabla 28. *Historia de usuario HU08.*

Número	**HU08**
Usuario	Usuario de la aplicación
Nombre de la historia	Gestión de la cuenta de usuario
Prioridad en el negocio	Media
Riesgo en desarrollo	Medio
Puntos estimados	5
Iteración asignada	5
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero gestionar la información de mi cuenta, de modo que pueda actualizar mi nombre y cambiar mi contraseña de forma segura desde la propia plataforma.
Validación	• Comprobar que el usuario pueda actualizar el nombre que se muestra en la plataforma. • Verificar que la contraseña pueda cambiarse validando la contraseña actual. • Corroborar que la nueva contraseña cumpla la longitud mínima establecida.

Fuente: Elaboración propia.

En la Tabla 29 se describe la activación del modo oscuro de la aplicación.

Tabla 29. *Historia de usuario HU09.*

Número	**HU09**
Usuario	Usuario de la aplicación
Nombre de la historia	Modo oscuro
Prioridad en el negocio	Baja
Riesgo en desarrollo	Bajo
Puntos estimados	2
Iteración asignada	6
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero alternar entre un tema claro y uno oscuro, con el fin de adaptar la apariencia de la interfaz a mis preferencias y a las condiciones de iluminación.
Validación	• Comprobar que el usuario pueda activar y desactivar el modo oscuro. • Verificar que la preferencia de tema se mantenga entre sesiones. • Corroborar que todos los componentes de la interfaz se visualicen correctamente en ambos temas.

Fuente: Elaboración propia.

En la Tabla 30 se describe el análisis integral de una parcela completa de cultivo.

Tabla 30. *Historia de usuario HU10.*

Número	**HU10**
Usuario	Usuario de la aplicación
Nombre de la historia	Análisis de parcela completa
Prioridad en el negocio	Alta
Riesgo en desarrollo	Alto
Puntos estimados	13
Iteración asignada	7
Programador responsable	Alex Ronaldo Luna Quilumba
Descripción	Como usuario de la aplicación, requiero delimitar sobre un mapa satelital el perímetro de una parcela y generar puntos de muestreo dentro del polígono para diagnosticar cada muestra con el modelo de inteligencia artificial, con el fin de obtener un diagnóstico fitosanitario integral del lote, con su severidad global, sin depender del análisis de una sola imagen.
Validación	• Comprobar que el usuario pueda dibujar el perímetro de la parcela marcando vértices sobre el mapa satelital hasta cerrar el polígono. • Verificar que el sistema calcule el área en hectáreas y distribuya automáticamente los puntos de muestreo dentro del polígono. • Constatar que cada muestra se diagnostique con el modelo de IA y presente su categoría (Sano, Tizón, Roya o Mancha Blanca) con su porcentaje de afectación. • Corroborar que el sistema agregue los resultados e indique la enfermedad dominante y la severidad global (Sano, Leve, Moderada o Grave). • Comprobar que el análisis completo se almacene en el historial y pueda consultarse en la pestaña de parcelas completas. • Verificar que, si se interrumpe el muestreo, el avance de la parcela se conserve y pueda retomarse posteriormente. • Corroborar que, ante una falla de conexión con el servicio de IA, el sistema notifique el error y permita reintentar la muestra sin perder el polígono trazado.

Fuente: Elaboración propia.

Product Backlog

Las historias de usuario definidas con anterioridad se ordenaron de acuerdo con su prioridad, su estimación y el sprint en el que serían desarrolladas. Este ordenamiento sirvió como guía de trabajo

para levantar de forma incremental los módulos de la aplicación web, atendiendo en primer lugar las funcionalidades imprescindibles para que el sistema operara. En la Tabla 31 se organiza la planificación inicial de las historias de usuario por sprint.

Tabla 31. *Product Backlog de la aplicación web.*

ID	Historia de usuario	Prioridad	Puntos	Sprint
HU01	Registro e inicio de sesión	Alta	5	SP01
HU07	Página principal	Media	3	SP05
HU02	Carga de imágenes del cultivo	Alta	3	SP02
HU03	Diagnóstico con inteligencia artificial	Alta	8	SP02
HU04	Visualización del mapa de calor	Alta	5	SP03
HU05	Historial de análisis	Media	3	SP03
HU06	Información sobre enfermedades	Baja	2	SP04
HU08	Gestión de la cuenta de usuario	Media	5	SP05
HU09	Modo oscuro	Baja	2	SP06
HU10	Análisis de parcela completa	Alta	13	SP07

Fuente: Elaboración propia.

Arquitectura de la aplicación web

La aplicación web se organizó bajo un esquema cliente-servidor de tres capas, con la finalidad de separar la interfaz de usuario, la lógica de procesamiento y el almacenamiento de la información. En la capa de presentación se situó el frontend construido en React, responsable de la interacción con el usuario, la carga de imágenes y la muestra de los resultados; la capa de lógica quedó a cargo del backend desarrollado con FastAPI, que recibe las imágenes, carga el modelo MiT-B0 y ejecuta la inferencia para producir el diagnóstico y el mapa de calor; y la capa de datos se conformó con una base de datos PostgreSQL, en la que se guardan los usuarios y el historial de análisis. El frontend se alojó en la plataforma Render y la base de datos en Supabase, en tanto que el backend con el modelo se desplegó en Hugging Face Spaces. En la Figura 21 se presenta la arquitectura general del sistema.

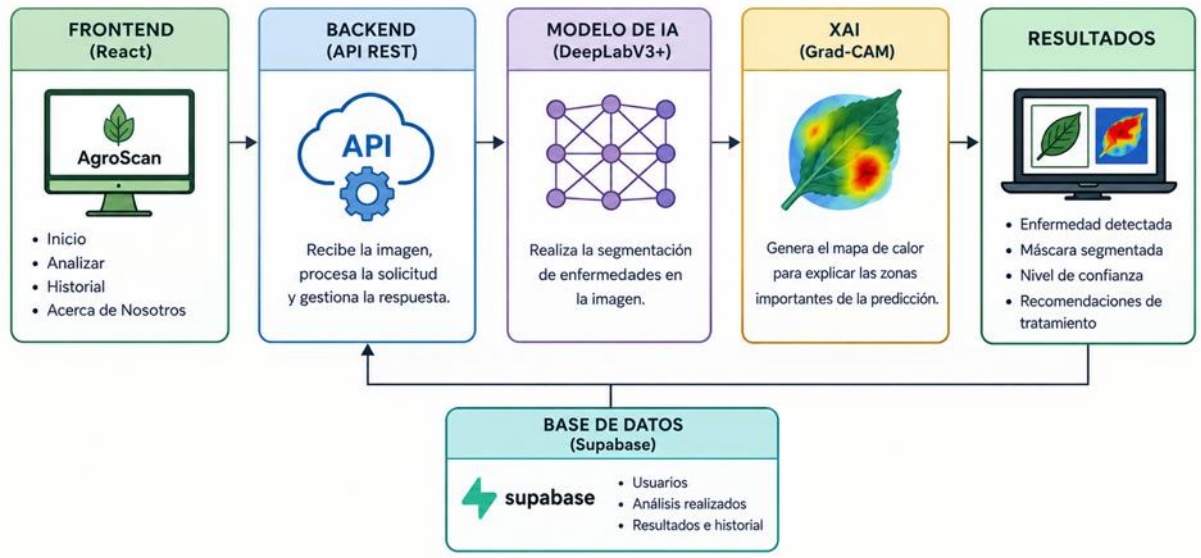


Figura 21. *Arquitectura de la aplicación web.*

Fuente: Elaboración propia.

3.7.3 Fase de Juego: desarrollo por sprints

En esta etapa se llevó a cabo la implementación de la aplicación web mediante iteraciones sucesivas de desarrollo. Cada sprint reunió un grupo de historias de usuario ligadas a funcionalidades concretas, lo que permitió edificar el sistema de manera progresiva; en cada iteración se partió de un objetivo de sprint, se planificaron las actividades correspondientes, se generó el incremento previsto y se revisaron los criterios de aceptación.

Planificación general de sprints

La planificación general permitió repartir las historias de usuario en siete sprints, atendiendo a su prioridad, complejidad y la dependencia entre las funcionalidades. En la Tabla 32 se presenta la distribución del trabajo por sprint y su vínculo con las historias de usuario.

Tabla 32. *Distribución general de sprints del proyecto.*

Sprint	Trabajo planificado	Historias vinculadas
SP01	Registro, autenticación y control de acceso	HU01
SP02	Carga de imágenes y diagnóstico mediante inteligencia artificial	HU02, HU03
SP03	Visualización del mapa de calor e historial de análisis	HU04, HU05
SP04	Sección informativa de enfermedades	HU06
SP05	Página principal y gestión de la cuenta de usuario	HU07, HU08
SP06	Modo oscuro de la interfaz	HU09
SP07	Análisis integral de la parcela completa	HU10

Fuente: Elaboración propia.

Con el fin de garantizar la calidad de los incrementos obtenidos en cada sprint, se estableció un criterio de terminado aplicable a las historias de usuario implementadas. En la Tabla 33 se establecen los criterios empleados para validar la finalización de cada incremento.

Tabla 33. *Definición de terminado aplicada durante los sprints.*

Criterio	Descripción
Funcionalidad implementada	La historia de usuario se implementó de acuerdo con el alcance fijado para el sprint correspondiente.
Validación de datos	Se revisaron las entradas y acciones del usuario para impedir registros incompletos o inconsistentes.
Persistencia y procesamiento correcto	La información se almacenó, consultó o procesó según la funcionalidad desarrollada.
Navegación e interfaz revisadas	La interfaz mantuvo el diseño general de la aplicación y permitió navegar sin inconvenientes.
Criterios de aceptación verificados	La funcionalidad se contrastó con los criterios establecidos en la historia de usuario correspondiente.

Fuente: Elaboración propia.

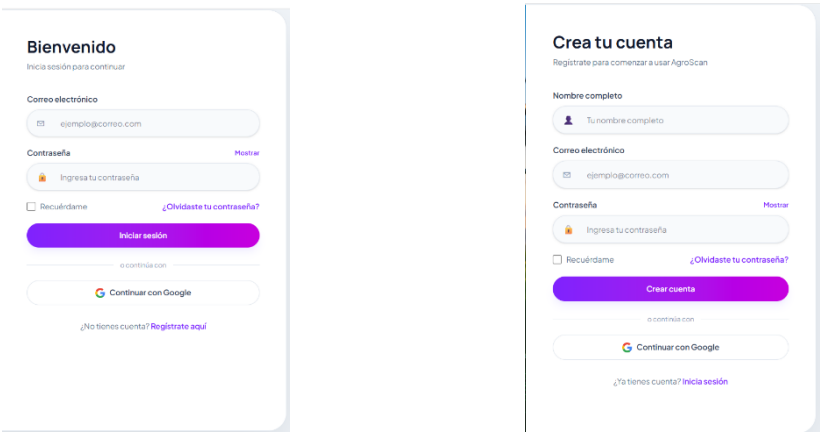
Objetivo del Sprint: El objetivo de este Sprint fue disponer del módulo de autenticación de la aplicación, que permitiera el registro de nuevos usuarios y el inicio de sesión con control de acceso, estableciendo la base de seguridad sobre la cual operarían las funcionalidades posteriores. El Sprint Backlog se detalla en la Tabla 34 y la revisión de los criterios de aceptación en la Tabla 35.

Tabla 34. *Sprint Backlog – Sprint 1.*

Historia	Nombre	Actividad desarrollada
HU01	Registro e inicio de sesión	Desarrollo de las pantallas de registro e inicio de sesión y de la lógica de autenticación segura apoyada en FastAPI y PostgreSQL.

Fuente: Elaboración propia.

Incremento desarrollado: como producto de esta iteración se consiguió el flujo inicial de acceso a la aplicación, de modo que el usuario puede crear una cuenta, autenticarse con su correo y contraseña e ingresar de forma protegida a las funcionalidades del sistema; adicionalmente, se incorporaron validaciones en los campos del registro y del inicio de sesión. El incremento obtenido se ilustra en la Figura 22.



Inicio de sesión (b) Registro de usuarios

Figura 22. *Interfaz de registro e inicio de sesión de la aplicación web.*

Fuente: Elaboración propia.

Tabla 35. *Revisión de criterios de aceptación – Sprint 1.*

Criterio de aceptación	Cumple
El usuario logra registrarse con su correo y contraseña.	Sí
El usuario ingresa con las credenciales previamente registradas.	Sí
Los datos del usuario quedan guardados en PostgreSQL.	Sí
El acceso a las funciones protegidas se gestiona de forma correcta.	Sí
Los campos obligatorios se validan antes de avanzar.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregó el módulo de registro e inicio de sesión (HU01), que cumplió con los criterios de aceptación definidos. El Product Owner validó que el usuario podía registrarse, autenticarse y acceder de forma controlada a la aplicación, por lo que el incremento se aceptó como terminado y quedó listo como base para el siguiente Sprint.

Retrospectiva del Sprint: La principal dificultad estuvo en la gestión segura de las credenciales y de las sesiones de usuario; se resolvió adoptando un esquema de autenticación basado en tokens con validaciones del lado del servidor. El equipo aprendió a estandarizar la validación de formularios, lo que agilizó el desarrollo de las interfaces en el Sprint siguiente.

Objetivo del Sprint: El objetivo de este Sprint fue habilitar la carga de imágenes del cultivo y su diagnóstico mediante el modelo de inteligencia artificial, de modo que el usuario pudiera obtener la identificación de la enfermedad a partir de una imagen. El Sprint Backlog se detalla en la Tabla 36 y la revisión de los criterios de aceptación en la Tabla 37.

Tabla 36. *Sprint Backlog – Sprint 2.*

Historia	Nombre	Actividad desarrollada
HU02	Carga de imágenes del cultivo	Construcción del flujo para cargar y validar una imagen obtenida con dron.
HU03	Diagnóstico con inteligencia artificial	Incorporación del modelo MiT-B0 en el backend para procesar la imagen y presentar el resultado del diagnóstico.

Fuente: Elaboración propia.

Incremento desarrollado: esta iteración dio como resultado el módulo de diagnóstico con inteligencia artificial, que permite al usuario cargar una imagen, comprobarla antes del análisis y procesarla con el modelo, tras lo cual el sistema muestra la enfermedad identificada y el porcentaje de área afectada. El incremento obtenido se ilustra en la Figura 23.

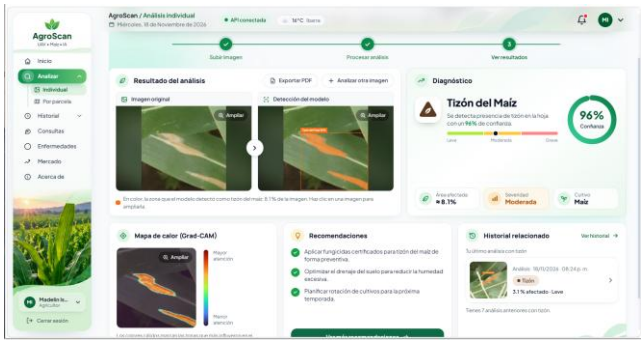


Figura 23. *Pantallas de análisis de imágenes.*

Fuente: Elaboración propia.

Tabla 37. *Revisión de criterios de aceptación – Sprint 2.*

Criterio de aceptación	Cumple
El usuario consigue subir una imagen del cultivo.	Sí
La aplicación comprueba la imagen antes de iniciar el análisis.	Sí
El modelo de inteligencia artificial procesa la imagen desde el backend.	Sí
El usuario observa la enfermedad detectada y el área afectada.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregaron la carga de imágenes del cultivo (HU02) y el diagnóstico con inteligencia artificial (HU03), cumpliendo los criterios de aceptación establecidos. El Product Owner verificó que el usuario podía subir una imagen y recibir el diagnóstico correspondiente del modelo, por lo que el incremento quedó aprobado y disponible para las funcionalidades de visualización del siguiente Sprint.

Retrospectiva del Sprint: La dificultad más relevante fue la integración del modelo de inteligencia artificial con el servidor y los tiempos de respuesta de la inferencia; se resolvió ajustando el preprocesamiento de las imágenes y el manejo asíncrono de las peticiones. El equipo aprendió a definir contratos claros entre el servicio de inteligencia artificial y la aplicación, lo que facilitó las integraciones posteriores.

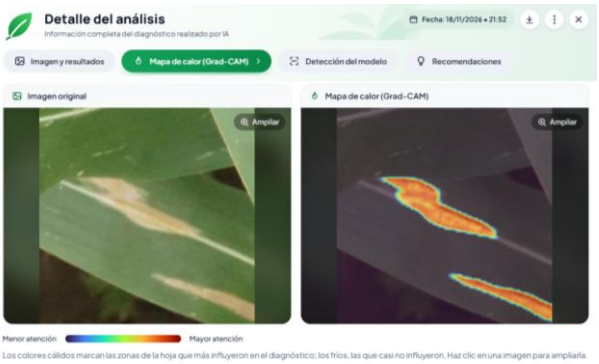
Objetivo del Sprint: El objetivo de este Sprint fue ofrecer la visualización del mapa de calor sobre las imágenes diagnosticadas y el historial de análisis del usuario, para que este pudiera interpretar los resultados y consultar sus diagnósticos previos. El Sprint Backlog se detalla en la Tabla 38 y la revisión de los criterios de aceptación en la Tabla 39.

Tabla 38. *Sprint Backlog – Sprint 3.*

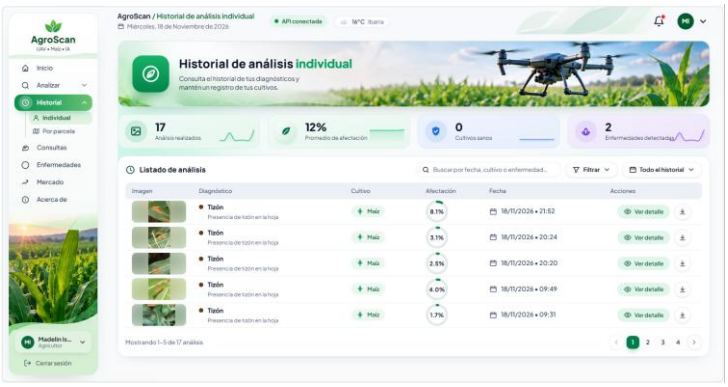
Historia	Nombre	Actividad desarrollada
HU04	Visualización del mapa de calor	Desarrollo de la generación y visualización del mapa de calor con Grad-CAM sobre la imagen analizada.
HU05	Historial de análisis	Elaboración del módulo de historial para guardar y consultar los análisis efectuados por el usuario.

Fuente: Elaboración propia.

Incremento desarrollado: de esta iteración surgieron la visualización del mapa de calor y el módulo de historial, con lo cual el usuario puede observar las zonas afectadas resaltadas sobre la imagen y revisar los análisis efectuados con anterioridad, acompañados de la información principal de cada resultado. El incremento obtenido se ilustra en la Figura 24.



Mapa de calor generado



Interfaz del historial de la aplicación web.

Figura 24. *Pantallas del mapa de calor y del historial de análisis.*

Fuente: Elaboración propia.

Tabla 39. *Revisión de criterios de aceptación – Sprint 3.*

Criterio de aceptación	Cumple
La aplicación produce y despliega el mapa de calor sobre la imagen analizada.	Sí
Las regiones resaltadas coinciden con las áreas afectadas.	Sí
Los análisis quedan almacenados y vinculados al usuario correspondiente.	Sí
El usuario logra revisar el historial de los análisis efectuados.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregaron la visualización del mapa de calor (HU04) y el historial de análisis (HU05), que satisficieron los criterios de aceptación definidos. El Product Owner

validó que el usuario podía visualizar las zonas afectadas y acceder a sus análisis anteriores, por lo que el incremento se aceptó y quedó listo para el siguiente Sprint.

Retrospectiva del Sprint: La dificultad principal fue la representación clara del mapa de calor sobre la imagen original y la organización del historial; se resolvió superponiendo la capa de calor con transparencia y paginando los registros almacenados. El equipo aprendió a reutilizar los componentes de visualización, lo que redujo el esfuerzo de desarrollo en las pantallas posteriores.

Objetivo del Sprint: El objetivo de este Sprint fue construir la sección informativa de la aplicación, que presentara al usuario la descripción de las enfermedades del maíz contempladas en el proyecto de forma clara y comprensible. El Sprint Backlog se detalla en la Tabla 40 y la revisión de los criterios de aceptación en la Tabla 41.

Tabla 40. *Sprint Backlog – Sprint 4.*

Historia	Nombre	Actividad desarrollada
HU06	Información sobre enfermedades	Desarrollo de la sección informativa con la descripción de las enfermedades del maíz contempladas en el proyecto.

Fuente: Elaboración propia.

Incremento desarrollado: como resultado de esta iteración se obtuvo la sección informativa de la aplicación, la cual permite al usuario consultar datos sobre las enfermedades del maíz contempladas en el proyecto, expuestos de forma clara y comprensible dentro de la interfaz. El incremento obtenido se ilustra en la Figura 25.

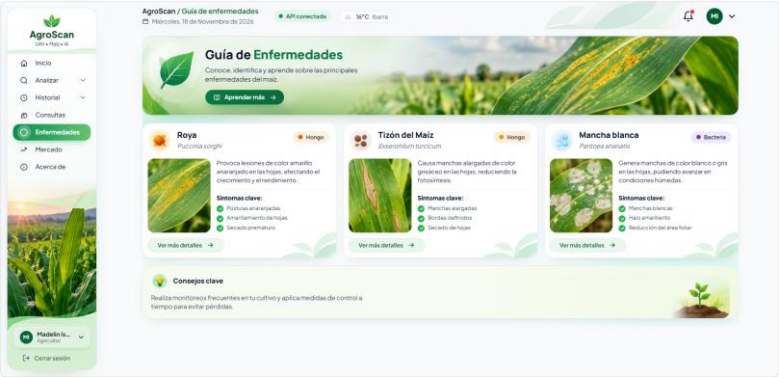


Figura 25. *Pantalla de la sección informativa sobre enfermedades.*

Fuente: Elaboración propia.

Tabla 41. *Revisión de criterios de aceptación – Sprint 4.*

Criterio de aceptación	Cumple
El usuario puede ingresar a la sección informativa.	Sí
La sección detalla las enfermedades contempladas en el proyecto.	Sí
La información se presenta de manera clara y comprensible.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregó la sección informativa sobre enfermedades (HU06), cumpliendo los criterios de aceptación definidos. El Product Owner verificó que el usuario podía consultar la información de cada enfermedad de manera organizada, por lo que el incremento quedó aprobado y disponible para el siguiente Sprint.

Retrospectiva del Sprint: La dificultad estuvo en estructurar el contenido técnico de las enfermedades de manera accesible para el usuario final; se resolvió estandarizando fichas descriptivas con un formato uniforme. El equipo aprendió a separar el contenido de su presentación, lo que facilitó la incorporación de nuevas pantallas informativas.

Objetivo del Sprint: El objetivo de este Sprint fue obtener la página principal de la aplicación y el módulo de gestión de la cuenta de usuario, de modo que el usuario accediera de forma organizada a las funcionalidades del sistema y administrara los datos de su perfil. El Sprint Backlog se detalla en la Tabla 42 y la revisión de los criterios de aceptación en la Tabla 43.

Tabla 42. *Sprint Backlog – Sprint 5.*

Historia	Nombre	Actividad desarrollada
HU07	Página principal	Desarrollo de la página principal de la aplicación, con la presentación de los módulos disponibles y los accesos directos a las funcionalidades del sistema.
HU08	Gestión de la cuenta de usuario	Implementación del módulo de gestión de la cuenta de usuario, con la edición de los datos del perfil y el cierre de sesión.

Fuente: Elaboración propia.

Incremento desarrollado: como resultado de esta iteración se obtuvieron la página principal de la aplicación y el módulo de gestión de la cuenta de usuario, que permiten al usuario acceder de forma organizada a las funcionalidades del sistema y administrar los datos de su perfil. El incremento obtenido se ilustra en la Figura 26.



Figura 26. *Pantalla de la página principal y del módulo de gestión de la cuenta de usuario.*

Fuente: Elaboración propia.

Tabla 43. *Revisión de criterios de aceptación – Sprint 5.*

Criterio de aceptación	**Cumple**
El usuario visualiza la página principal con los módulos y accesos directos disponibles.	Sí
El usuario puede editar los datos de su perfil de usuario.	Sí
El usuario puede cerrar sesión de forma segura.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregaron la página principal (HU07) y la gestión de la cuenta de usuario (HU08), cumpliendo los criterios de aceptación definidos. El Product Owner validó que el usuario podía visualizar los módulos disponibles, editar su perfil y cerrar sesión, por lo que el incremento se aceptó y quedó listo para el siguiente Sprint.

Retrospectiva del Sprint: La dificultad fue mantener la coherencia de navegación entre los módulos ya existentes y los nuevos; se resolvió centralizando la estructura de navegación y la gestión del estado de la sesión. El equipo aprendió a consolidar los componentes de interfaz comunes, lo que agilizó los ajustes visuales posteriores.

Objetivo del Sprint: El objetivo de este Sprint fue incorporar el modo oscuro de la aplicación, permitiendo al usuario alternar el tema visual de la interfaz y conservar dicha preferencia entre sesiones. El Sprint Backlog se detalla en la Tabla 44 y la revisión de los criterios de aceptación en la Tabla 45.

Tabla 44. *Sprint Backlog – Sprint 6.*

Historia	Nombre	Actividad desarrollada
HU09	Modo oscuro	Implementación del modo oscuro de la aplicación, con el cambio de tema de la interfaz y la persistencia de la preferencia del usuario entre sesiones.

Fuente: Elaboración propia.

Incremento desarrollado: como resultado de esta iteración se incorporó el modo oscuro de la aplicación, que permite al usuario alternar el tema visual de la interfaz y conservar dicha preferencia entre sesiones. El incremento obtenido se ilustra en la Figura 27.

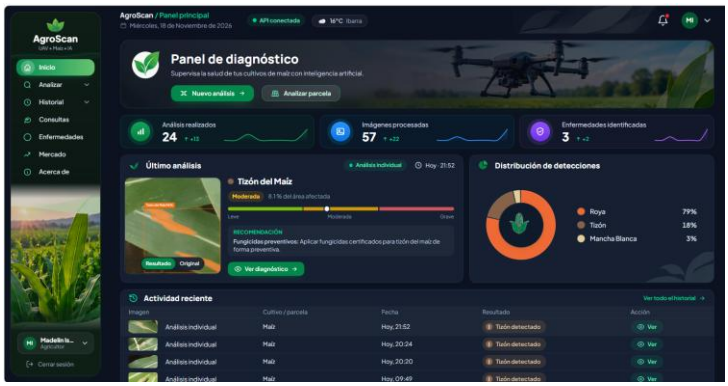


Figura 27. *Pantalla de la aplicación con el modo oscuro activado.*

Fuente: Elaboración propia.

Tabla 45. *Revisión de criterios de aceptación – Sprint 6.*

Criterio de aceptación	Cumple
El usuario puede activar y desactivar el modo oscuro.	Sí
La interfaz cambia de tema de forma coherente en todas las pantallas.	Sí
La preferencia de tema seleccionada se conserva entre sesiones.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregó el modo oscuro de la interfaz (HU09), cumpliendo los criterios de aceptación definidos. El Product Owner verificó que el usuario podía activar y desactivar el modo oscuro de forma coherente en todas las pantallas y que la preferencia se conservaba, por lo que el incremento quedó aprobado y listo para el siguiente Sprint.

Retrospectiva del Sprint: La dificultad fue garantizar el contraste y la legibilidad en todas las pantallas al cambiar de tema; se resolvió definiendo una paleta de colores centralizada mediante

variables de estilo. El equipo aprendió a aplicar los temas de forma global, lo que evitó ajustes repetitivos y preparó la base visual para el módulo final.

Objetivo del Sprint: El objetivo de este Sprint fue construir el módulo de análisis de parcela completa, que permitiera delimitar una parcela sobre un mapa satelital, generar puntos de muestreo, diagnosticar cada muestra con el modelo de inteligencia artificial y consolidar la severidad global del lote. El Sprint Backlog se detalla en la Tabla 46 y la revisión de los criterios de aceptación en la Tabla 47.

Tabla 46. *Sprint Backlog – Sprint 7.*

Historia	Nombre	Actividad desarrollada
HU10	Análisis de parcela completa	Desarrollo del módulo de análisis de parcela completa, con la delimitación del polígono sobre el mapa satelital, la generación de puntos de muestreo, el diagnóstico de cada muestra mediante inteligencia artificial y la consolidación de la severidad global del lote.

Fuente: Elaboración propia.

Incremento desarrollado: como resultado de esta iteración se construyó el módulo de análisis de parcela completa, que permite al usuario delimitar una parcela sobre un mapa satelital, generar puntos de muestreo, diagnosticar cada muestra con el modelo de inteligencia artificial y obtener la severidad global del lote. El incremento obtenido se ilustra en la Figura 28.

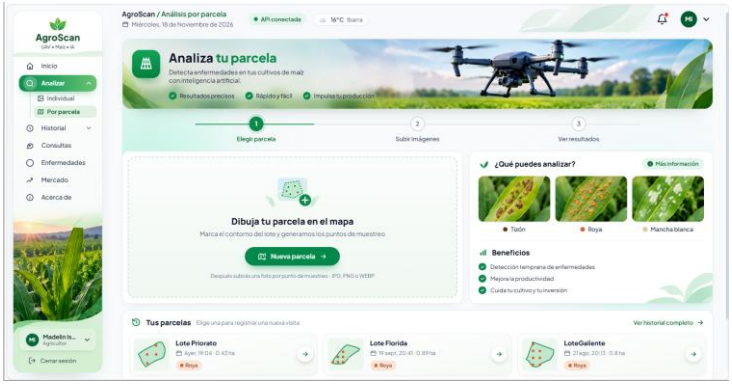


Figura 28. *Pantalla del módulo de análisis de parcela completa.*

Fuente: Elaboración propia.

Tabla 47. *Revisión de criterios de aceptación – Sprint 7.*

Criterio de aceptación	Cumple
El usuario puede delimitar el perímetro de la parcela sobre el mapa satelital.	Sí
El sistema genera los puntos de muestreo y diagnostica cada uno con el modelo de inteligencia artificial.	Sí
El sistema consolida la enfermedad dominante y la severidad global del lote.	Sí

Fuente: Elaboración propia.

Revisión del Sprint: En este Sprint se entregó el módulo de análisis de parcela completa (HU10), cumpliendo los criterios de aceptación definidos. El Product Owner validó que el usuario podía delimitar el perímetro de la parcela, generar los puntos de muestreo, obtener el diagnóstico de cada uno y consultar la severidad global del lote, por lo que el incremento se aceptó como versión final del producto.

Retrospectiva del Sprint: La dificultad fue coordinar el cálculo geométrico del polígono con la ejecución del modelo de inteligencia artificial sobre múltiples muestras; se resolvió distribuyendo los puntos de muestreo de forma controlada y consolidando los resultados de manera incremental. El equipo aprovechó la experiencia acumulada en los Sprints previos para integrar de forma robusta la cartografía, la inteligencia artificial y la visualización de resultados.

Matriz de trazabilidad Scrum

Concluidos los sprints de desarrollo, se efectuó una retrospectiva general del proceso Scrum seguido en la construcción de la aplicación web, revisión que permitió reconocer los progresos alcanzados, las dificultades enfrentadas y los ajustes introducidos durante la implementación gradual de los módulos, sobre todo en la integración del modelo de inteligencia artificial, la generación del mapa de calor y la persistencia del historial de análisis; de dicha retrospectiva se derivaron correcciones puntuales en la navegación, la estabilidad funcional y la presentación visual de los resultados.

A continuación, se consolidó una matriz de trazabilidad de Scrum con el propósito de enlazar los requerimientos funcionales con las historias de usuario, los sprints ejecutados, las evidencias producidas y las pruebas funcionales realizadas, lo que permitió comprobar la correspondencia entre la planificación inicial y los incrementos desarrollados a lo largo de la construcción de la aplicación web. En la Tabla 48 se relacionan los requerimientos, las historias, los sprints, las evidencias y las pruebas funcionales.

Tabla 48. *Matriz de trazabilidad Scrum de la aplicación web.*

RF	HU	Sprint	Evidencia	Prueba funcional	Estado
RF01	HU01	SP01	Pantallas de registro y autenticación de usuarios	Verificación del registro y del inicio de sesión.	Cumplido
RF02	HU02	SP02	Pantalla de carga de imágenes del cultivo	Comprobación de la carga y validación de la imagen.	Cumplido
RF03	HU03	SP02	Pantalla de diagnóstico con IA	Procesamiento de la imagen y generación del diagnóstico.	Cumplido
RF04	HU04	SP03	Vista del mapa de calor	Comprobación del mapa de calor generado con Grad-CAM.	Cumplido
RF05	HU05	SP03	Pantalla del historial de análisis	Consulta de los análisis previamente efectuados.	Cumplido
RF06	HU06	SP04	Pantalla de la sección informativa	Revisión de la información sobre enfermedades.	Cumplido
RF07	HU07	SP05	Pantalla de la página principal	Verificación del acceso a la página principal.	Cumplido
RF08	HU08	SP05	Pantalla de gestión de la cuenta	Comprobación de la edición del nombre y el cambio de contraseña.	Cumplido
RF09	HU09	SP06	Interfaz con modo oscuro	Verificación de la activación del modo oscuro.	Cumplido
RF10	HU10	SP07	Pantalla de análisis de parcela completa	Verificación del análisis integral de la parcela y su diagnóstico.	Cumplido

Fuente: Elaboración propia.

3.7.4 Fase de Post-Juego: pruebas, ajustes y despliegue

El Post-Juego constituyó la etapa de cierre del desarrollo de la aplicación web, en la cual se ejecutaron las pruebas funcionales, se corrigieron los errores detectados, se realizaron ajustes visuales y se preparó la versión definitiva para su despliegue y su posterior evaluación con usuarios.

Pruebas funcionales de la aplicación web

Las pruebas funcionales tuvieron como finalidad comprobar el comportamiento de los módulos construidos durante los sprints, verificando que las funcionalidades principales reaccionaran de forma

correcta ante las acciones del usuario y que la información se registrara, consultara y procesara de manera adecuada. En la Tabla 49 se resumen las pruebas aplicadas a los módulos principales de la aplicación.

Tabla 49. *Pruebas funcionales de la aplicación web.*

Módulo	Prueba realizada	Resultado
Autenticación	Registro y autenticación de los usuarios.	Correcto
Carga de imágenes	Subida y validación de las imágenes del cultivo.	Correcto
Diagnóstico IA	Procesamiento de la imagen y obtención del diagnóstico.	Correcto
Mapa de calor	Producción y despliegue del mapa de calor con Grad-CAM.	Correcto
Historial	Consulta de los análisis almacenados por el usuario.	Correcto
Información	Revisión de la sección informativa de enfermedades.	Correcto

Fuente: Elaboración propia.

Corrección de errores y ajustes finales

Tras la ejecución de las pruebas funcionales se detectaron errores y ajustes que debían atenderse en distintos módulos de la aplicación web, cuyas correcciones contribuyeron a reforzar la estabilidad del sistema, la autenticación de los usuarios, el procesamiento de las imágenes y la persistencia de los análisis realizados. En la Tabla 50 se resumen las correcciones aplicadas.

Tabla 50. *Correcciones y ajustes realizados en la aplicación web.*

Módulo	Ajuste realizado	Estado
Autenticación	Ajuste en el control de sesión de los usuarios.	Corregido
Carga de imágenes	Refuerzo de la validación del formato de las imágenes cargadas.	Corregido
Diagnóstico IA	Optimización del tiempo de respuesta en la inferencia del modelo.	Corregido
Mapa de calor	Arreglo de la superposición del mapa de calor sobre la imagen.	Corregido
Historial	Revisión del almacenamiento y la consulta de los análisis efectuados.	Corregido
Interfaz y navegación	Corrección de detalles visuales y de la transición entre pantallas.	Corregido

Fuente: Elaboración propia.

Despliegue y preparación para la evaluación con usuarios

Al concluir el desarrollo, la aplicación web quedó desplegada en un entorno de producción, con el frontend alojado en la plataforma Render, la base de datos PostgreSQL en Supabase y el backend junto con el modelo de inteligencia artificial desplegado en Hugging Face Spaces, de manera que el sistema quedó accesible desde cualquier navegador web sin requerir instalación adicional.

Una vez desplegada la aplicación y cerradas las correcciones finales, se dispuso la evaluación con usuarios con el fin de recoger su percepción acerca del funcionamiento, la utilidad y la facilidad de uso del sistema, tomando como referencia las características de usabilidad y eficiencia de la norma ISO/IEC 25010; tanto los resultados de esa evaluación como el detalle técnico del despliegue se exponen y analizan en el Capítulo IV.

Para consolidar el trabajo realizado bajo la metodología Scrum, en la Tabla 51 se resume la trazabilidad de cada requisito funcional con su historia de usuario, el sprint en que se implementó, el componente de la arquitectura involucrado, la prueba aplicada y el resultado esperado.

Tabla 51. *Resumen de trazabilidad entre requisito, historia de usuario, sprint, componente, prueba y resultado esperado.*

Requisito	Historia	Sprint	Componente	Prueba	Resultado esperado
RF01. Registro e inicio de sesión	HU01	SP01	Frontend, backend y base de datos	Verificación del registro y del inicio de sesión.	El usuario se registra, ingresa con sus credenciales y accede a las funciones protegidas.
RF02. Carga de imágenes	HU02	SP02	Frontend y backend	Comprobación de la carga y validación de la imagen.	La imagen del cultivo se carga y se valida antes del análisis.
RF03. Diagnóstico con IA	HU03	SP02	Backend y modelo MiT-B0	Procesamiento de la imagen y generación del diagnóstico.	Se presenta la enfermedad identificada y el porcentaje de área afectada.
RF04. Mapa de calor	HU04	SP03	Backend (Grad-CAM) y frontend	Comprobación del mapa de calor generado con Grad-CAM.	El mapa de calor se superpone a la imagen y resalta las regiones afectadas.
RF05. Historial de análisis	HU05	SP03	Frontend, backend y base de datos	Consulta de los análisis previamente efectuados.	Los análisis quedan almacenados y pueden consultarse por cada usuario.
RF06. Información sobre enfermedades	HU06	SP04	Frontend	Revisión de la información sobre enfermedades.	La sección describe de forma clara las enfermedades contempladas en el proyecto.
RF07. Página principal	HU07	SP05	Frontend	Verificación del acceso a la página principal.	Tras iniciar sesión se muestran los módulos y los accesos directos.
RF08. Gestión de la cuenta	HU08	SP05	Frontend, backend y base de datos	Comprobación de la edición del nombre y el cambio de contraseña.	El nombre se actualiza y la contraseña se cambia validando la actual.
RF09. Modo oscuro	HU09	SP06	Frontend	Verificación de la activación del modo oscuro.	El tema cambia de forma coherente y se conserva entre sesiones.
RF10. Análisis de parcela completa	HU10	SP07	Frontend, backend y modelo MiT-B0	Verificación del análisis integral de la parcela y su diagnóstico.	Se consolidan la enfermedad dominante y la severidad global del lote.

Fuente: Elaboración propia.

CAPÍTULO IV

4. RESULTADOS Y ANÁLISIS

En este capítulo se presentan y analizan los resultados obtenidos a lo largo del desarrollo del proyecto, abarcando las distintas etapas que componen la solución propuesta, desde la construcción del conjunto de datos hasta la puesta en producción del sistema. En primer lugar se describe el dataset final construido a partir de imágenes capturadas con dron y de fuentes públicas; a continuación se presentan los resultados del modelo de segmentación semántica de enfermedades en cultivos de maíz, partiendo de las métricas de evaluación y de los resultados del entrenamiento, continuando con la evaluación cuantitativa por clase, el análisis de la matriz de confusión y la comparación entre las distintas arquitecturas evaluadas, para luego interpretar el comportamiento del modelo mediante los mapas de calor generados con inteligencia artificial explicable; posteriormente se presentan los resultados de la aplicación web desarrollada y los de su despliegue en un entorno de producción, y finalmente se discuten los hallazgos frente a los objetivos planteados y a los trabajos relacionados, de modo que se pueda valorar de forma objetiva el desempeño alcanzado por el sistema propuesto.

4.1 Resultados de la construcción del dataset

El primer resultado del proyecto corresponde al conjunto de datos construido desde cero para el entrenamiento del modelo de segmentación, el cual se conformó a partir de tres fuentes complementarias: imágenes propias capturadas con un dron DJI Mini 3 en cultivos de maíz de la comunidad de Pogllocunga, el dataset público Corn Disease Drone Images de Kaggle [7] y las imágenes aéreas del estudio "Quantitative Phenotyping of Northern Leaf Blight in UAV Images" de Stewart et al. [35] de modo que se logró reunir un conjunto representativo de las tres enfermedades consideradas: tizón, roya y mancha blanca.

Tras la consolidación de las distintas fuentes se superaron las 3000 imágenes, en las que la enfermedad del tizón fue la predominante por provenir de los datasets públicos, mientras que la roya y la mancha blanca se obtuvieron principalmente de las capturas propias; posteriormente, las imágenes fueron anotadas en la plataforma Roboflow mediante máscaras de segmentación para las tres clases de enfermedad, además de la clase Fondo, y se exportó el conjunto en formato COCO para su uso en el entrenamiento.

Para preparar el conjunto definitivo, las imágenes propias se recortaron a una resolución de 384 × 384 píxeles; el conjunto anotado se dividió en 80 % para entrenamiento, 10 % para validación y 10 % para prueba, y sobre la partición de entrenamiento se aplicó un aumento de datos (×3) con Roboflow mediante volteos, rotaciones, recortes, zoom y ajustes de exposición, sin aplicar transformaciones adicionales a los conjuntos de validación y prueba para conservar imágenes en su estado original. En la Tabla 52 se resume el conjunto de datos final construido.

Tabla 52. *Resumen del dataset final construido.*

Aspecto	Valor
Fuentes de datos	UAV + Kaggle Corn Disease Drone Images + Northern Leaf Blight UAV
Clases de enfermedad	Tizón, Roya, Mancha blanca
Clase adicional	Fondo
Imágenes consolidadas	3164
Imágenes anotadas y divididas	1256
Entrenamiento (80 %)	1005
Validación (10 %)	126
Prueba (10 %)	125
Resolución de exportación	384 × 384 px
Aumento de datos (Roboflow ×3, solo entrenamiento)	3015 entrenamiento (1005 × 3) + 126 validación + 125 prueba = 3266 imágenes exportadas
Formato de anotación	COCO (máscaras de segmentación)
Imágenes por clase — Entrenamiento (×3, n=3015)	Tizón: 2406 (79.8 %) · Roya: 348 (11.5 %) · Mancha blanca: 261 (8.7 %)
Imágenes por clase — Validación (n=126)	Tizón: 95 (75.4 %) · Roya: 20 (15.9 %) · Mancha blanca: 11 (8.7 %)
Imágenes por clase — Prueba (n=125)	Tizón: 100 (80.0 %) · Roya: 12 (9.6 %) · Mancha blanca: 13 (10.4 %)
Nota sobre la clase Fondo	No se reporta como clase predominante por imagen, ya que está presente como segmento de fondo en prácticamente la totalidad de las imágenes de las tres particiones.

Fuente: Elaboración propia.

Como se observa en la Tabla 52, el conjunto final quedó conformado por 1256 imágenes anotadas, de las cuales 1005 se destinaron al entrenamiento, 126 a la validación y 125 a la prueba, alcanzando 3266 imágenes en total tras el aumento de datos con Roboflow (3015 de entrenamiento, 126 de validación y 125 de prueba); este equilibrio entre fuentes propias y públicas, junto con el aumento aplicado exclusivamente al conjunto de entrenamiento, permitió contar con un dataset variado y representativo de las condiciones reales de campo, que constituye la base sobre la cual se entrenaron y evaluaron las distintas arquitecturas de segmentación. El desglose por clase que se añade al final de la Tabla 52 confirma que las tres enfermedades están representadas en cada partición, con el tizón como clase predominante (79.8–80.0 % de las imágenes) por su origen mayoritario en los datasets públicos, y la roya y la mancha blanca en proporciones minoritarias pero presentes en entrenamiento, validación

y prueba; dado que, como se detalla en la sección 3.3.3, la partición 80/10/10 se realizó antes de aplicar el aumento de datos, no existe solapamiento de imágenes entre los tres conjuntos.

4.2 Evaluación del modelo de inteligencia artificial

4.2.1 Diseño experimental para la evaluación del modelo

El modelo de segmentación se evaluó con dos objetivos. El primero fue comprobar qué tan bien detecta las enfermedades del maíz. El segundo fue respaldar las decisiones tomadas al construirlo. Para ello, la evaluación se organizó en tres partes. Se compararon distintas arquitecturas de codificador bajo la misma configuración. Se analizó la sensibilidad del modelo a la resolución y se realizó un estudio de ablación. Finalmente se describió con mayor detalle el modelo seleccionado. En esa descripción se tomaron en cuenta su exactitud, su comportamiento por clase, su interpretabilidad y su eficiencia.

Todas las pruebas se realizaron sobre el mismo conjunto de datos del Capítulo III. Este conjunto reúne las imágenes capturadas con el dron y las obtenidas de repositorios públicos. Todas fueron anotadas píxel por píxel en cuatro clases: Fondo, Tizón, Roya y Mancha Blanca. El conjunto se dividió aleatoriamente en tres partes: 80 % para entrenamiento, 10 % para validación y 10 % para prueba. La validación cumplió una doble función: sirvió para controlar el entrenamiento y decidir cuándo detenerlo, y también para comparar arquitecturas y resoluciones en igualdad de condiciones. El conjunto de prueba, en cambio, se mantuvo completamente aislado del entrenamiento y de la validación, y se utilizó únicamente para reportar el desempeño final del modelo seleccionado en datos nunca antes vistos, cuyos resultados se presentan en la sección 4.2.4.

El desempeño se calculó en el conjunto de validación. Se usaron los pesos obtenidos con la media móvil exponencial (EMA). También se aplicó aumento en el tiempo de prueba (TTA) con las ocho vistas del grupo diédrico D4. Las métricas fueron las mismas que se definieron en la sección 3.5.1: el mIoU, el IoU por clase, la precisión, la exhaustividad, el F1, la precisión promedio (mAP) y la exactitud a nivel de píxel. La comparación entre los modelos se hizo en igualdad de condiciones. Tanto las arquitecturas de la sección 4.2.2 como las variantes de resolución yción de la sección 4.2.3 se entrenaron y evaluaron de la misma manera. La resolución de entrada fue de 384 × 384 píxeles. La función de pérdida fue la híbrida por fases. El optimizador fue AdamW con calentamiento y recocido por coseno. El entrenamiento tuvo un máximo de 70 épocas con parada temprana con paciencia de 20. Se mantuvo la misma división de los datos para todas las comparaciones, sin fijar una semilla aleatoria explícita.

Todas las pruebas se ejecutaron en Google Colab Pro. Se utilizó una GPU NVIDIA con el framework PyTorch y la librería segmentation_models_pytorch. Para aprovechar la memoria y acelerar el cómputo, se aplicó entrenamiento en precisión mixta (AMP) con un lote de 10 imágenes. En cada prueba también se registraron métricas de eficiencia. Estos fueron el número de parámetros, el tamaño

del modelo y la velocidad de inferencia. Estos datos permiten saber qué tan viable es integrar el modelo en la aplicación web. Ese tema se aborda en la sección 4.2.8.

4.2.2 Comparación de arquitecturas evaluadas

Una vez definido el diseño experimental, el primer eje de la evaluación consistió en comparar distintas arquitecturas de codificador para elegir la que mejor se ajusta a las necesidades de este trabajo, ya que la elección del codificador es la que más influye en el equilibrio entre la exactitud de la segmentación y el costo computacional del modelo. Para que la comparación fuera justa, se mantuvo fija toda la configuración descrita en la sección 4.2.1: la misma cabeza de segmentación DeepLabV3+, la misma resolución de entrada de 384 × 384 píxeles, la misma función de pérdida, el mismo optimizador y la misma división 80/10/10 del dataset, de modo que la única diferencia entre un experimento y otro fuera el codificador utilizado. De esta forma se entrenaron y evaluaron siete codificadores que van desde variantes muy livianas hasta variantes más pesadas, abarcando tanto redes puramente convolucionales como codificadores de tipo transformer, y en la Tabla 53 se resumen sus resultados en cuanto a número de parámetros, tamaño del modelo, costo computacional (GFLOPs), desempeño de segmentación y velocidad de inferencia.

Tabla 53. *Comparación del desempeño de las arquitecturas de codificador evaluadas sobre el conjunto de validación.*

Codificador	Parámetros	Tamaño (MB)	GFLOPs	mIoU	F1 macro	Píxel Accuracy	FPS
MiT-B0	4 136 820	15.78	7.72	0.8182	0.8972	0.9767	70.4
MobileNetV2	4 379 284	16.71	6.68	0.8132	0.8938	0.9760	109.0
EfficientNet-B0	4 908 240	18.72	6.37	0.7955	0.8819	0.9740	72.5
MobileOne-S0	6 090 244	23.23	10.33	0.8191	0.8975	0.9764	32.0
ResNet18	12 330 068	47.04	20.54	0.8385	0.9099	0.9790	182.6
MiT-B1	14 304 980	54.57	21.37	0.8184	0.8973	0.9771	69.4
MiT-B2	25 349 844	96.70	34.87	0.8144	0.8947	0.9759	39.5

Fuente: Elaboración propia. GFLOPs calculados para una imagen de entrada de 384 × 384 píxeles (1 MAC = 2 FLOPs). FPS medidos en GPU sin TTA.

Los resultados de la Tabla 53 muestran que las diferencias de exactitud entre los siete codificadores son relativamente pequeñas, ya que todos alcanzan un mIoU superior a 0.79 y un F1 macro por encima de 0.88, lo que indica que la cabeza DeepLabV3+ y el resto de la configuración aportan una base sólida sin importar el codificador que se use. Aun así, se observa que los codificadores más pesados obtienen

los mejores valores en términos absolutos, siendo ResNet18 el de mayor mIoU (0.8385) y F1 macro (0.9099), aunque alcanza 47.04 MB y más de 12 millones de parámetros. Entre los codificadores de tipo Transformer, MiT-B1 logra el mejor desempeño (mIoU de 0.8184), muy cerca de MiT-B2 (0.8144), pero este último llega a los 96.70 MB y a más de 25 millones de parámetros, lo que lo vuelve poco práctico para un despliegue liviano. En el otro extremo, los codificadores livianos como MiT-B0, MobileNetV2 y EfficientNet-B0 se mantienen por debajo de los 19 MB y, pese a ello, conservan un desempeño muy competitivo, con apenas dos o tres puntos porcentuales de diferencia frente a los modelos grandes. En cuanto al costo computacional, MiT-B0 requiere 7.72 GFLOPs por imagen de 384 × 384 píxeles, entre 2.7 y 4.5 veces menos que ResNet18 (20.54), MiT-B1 (21.37) y MiT-B2 (34.87); solo MobileNetV2 (6.68) y EfficientNet-B0 (6.37) realizan menos operaciones, pero con un mIoU inferior al de MiT-B0 (0.8132 y 0.7955 frente a 0.8182).

Tomando en cuenta este equilibrio, se eligió el codificador MiT-B0 como base del modelo final, ya que es el más liviano de todos con 4.14 millones de parámetros y 15.78 MB, mantiene un mIoU de 0.8182 y un F1 macro de 0.8972 que lo sitúan muy cerca de codificadores considerablemente más grandes y, al ser un codificador de tipo Transformers con ciertas características de una cnn, se alinea con el enfoque híbrido CNN-Transformer que se planteó como eje de este trabajo. Si bien MobileNetV2 ofrece una velocidad de inferencia mayor (109.0 frente a 70.4 FPS) y ResNet18 alcanza la mayor velocidad de todas (182.6 FPS), ambos son codificadores puramente convolucionales y, en el caso de ResNet18, requieren casi el triple de tamaño en disco, por lo que MiT-B0 resulta la opción que mejor combina un tamaño reducido, una exactitud alta y la capacidad de captar el contexto global de las imágenes, características que resultan claves para poder integrar el modelo dentro de la aplicación web tal como se analiza en la sección 4.2.8.

Dado que la validación se utilizó para ajustar y seleccionar cada modelo, la comparación se completó con el conjunto de prueba, formado por imágenes independientes del entrenamiento. En la Tabla 54 se presentan los resultados de los siete codificadores en validación y en prueba.

Tabla 54. *Comparación de los codificadores en validación y prueba.*

Codificador	mIoU validación	F1 macro validación	mIoU prueba	F1 macro prueba
MiT-B0	0.8182	0.8972	0.7593	0.8560
MobileNetV2	0.8132	0.8938	0.7588	0.8559
EfficientNet-B0	0.7955	0.8819	0.7361	0.8390
MobileOne-S0	0.8191	0.8975	0.7495	0.8487
ResNet18	0.8385	0.9099	0.7528	0.8521
MiT-B1	0.8184	0.8973	0.7674	0.8619
MiT-B2	0.8144	0.8947	0.7655	0.8611

Fuente: Elaboración propia. Las métricas de prueba se obtuvieron con los pesos EMA y TTA.

En el conjunto de prueba, los tres codificadores de tipo Transformer (MiT-B1, MiT-B2 y MiT-B0) obtienen los mayores mIoU, por delante de las cuatro redes convolucionales. ResNet18, que ocupa el primer lugar en validación (0.8385), cae al quinto puesto en prueba (0.7528) y presenta la mayor caída entre ambos conjuntos (Δ de −0.0857, frente a −0.0589 de MiT-B0), lo que indica que su ventaja en validación no se mantiene ante imágenes nuevas. MiT-B1 es el más exacto en prueba (0.7674), pero MiT-B0 queda a solo 0.8 puntos de mIoU (0.7593) con 3.5 veces menos parámetros (4.14 M frente a 14.30 M) y 15.78 MB frente a 54.57 MB. Frente a ResNet18, MiT-B0 logra un mIoU mayor en prueba con casi tres veces menos parámetros (4.14 M frente a 12.33 M), tamaño (15.78 MB frente a 47.04 MB) y operaciones (7.72 frente a 20.54 GFLOPs); si bien ResNet18 es más rápido (182.6 frente a 70.4 FPS), los 70.4 FPS de MiT-B0 ya superan lo necesario para un uso interactivo, como se muestra en la sección 4.2.8. Es, además, el codificador con menos parámetros y menor tamaño de los siete y el más exacto en prueba entre los modelos de menos de 20 MB, aunque con una ventaja mínima sobre MobileNetV2 (0.7593 frente a 0.7588). En conjunto, MiT-B0 ofrece el mejor equilibrio entre exactitud, generalización y costo para ejecutarse en la aplicación web.

La selección de MiT-B0 frente al resto de alternativas responde a lo siguiente. Primero, supera a EfficientNet-B0 en las cuatro métricas de la tabla (0.7593 frente a 0.7361 de mIoU en prueba) y a MobileNetV2 en todas ellas, aunque por márgenes mínimos en prueba. Segundo, MobileOne-S0 iguala a MiT-B0 en validación (0.8191 frente a 0.8182 de mIoU), pero queda 0.0098 por debajo en prueba (0.7495) y requiere 6.09 M de parámetros, 23.23 MB y 10.33 GFLOPs, con solo 32.0 FPS. Tercero, MiT-B2 no mejora a MiT-B0 en validación (0.8144 frente a 0.8182) y MiT-B1 lo supera en prueba por menos de un punto de mIoU y de F1 macro, a cambio de 6.1 y 3.5 veces más parámetros, respectivamente. Por último, la ventaja de ResNet18 en validación (0.8385 frente a 0.8182 de mIoU) no se conserva en prueba. Por ello, MiT-B0 se adoptó como codificador del modelo final: combina un desempeño competitivo y estable ante imágenes nuevas con el menor número de parámetros y tamaño de los siete y, al ser un codificador de tipo Transformer, es coherente con el enfoque híbrido CNN-Transformers del trabajo.

4.2.3 Sensibilidad del modelo a la resolución de entrada

El segundo eje de la evaluación consistió en analizar la sensibilidad del codificador MiT-B0 al tamaño de la imagen de entrada, evaluando la configuración completa en cuatro resoluciones: 256 × 256, 384 × 384, 512 × 512 y 768 × 768 píxeles. La Tabla 55 resume los resultados obtenidos en el conjunto de validación.

Tabla 55. *Sensibilidad del codificador a la resolución de entrada sobre el conjunto de validación.*

Resolución (px)	mIoU	F1 macro	Tiempo/época (s)	FPS
256 × 256	0.8151	0.8949	24.8	68.7
384 × 384	0.8182	0.8972	28.0	70.4
512 × 512	0.8087	0.8912	35.3	73.2
768 × 768	0.8016	0.8862	59.2	66.3

Fuente: Elaboración propia.

La Tabla 55, calculada sobre el conjunto de validación, muestra que la resolución de 384 × 384 píxeles obtiene el mejor desempeño de las cuatro evaluadas, con un mIoU de 0.8182 y un F1 macro de 0.8972, por encima de 256 píxeles (0.8151 / 0.8949), 512 píxeles (0.8087 / 0.8912) y 768 píxeles (0.8016 / 0.8862). A diferencia de lo que suele observarse en tareas de segmentación, aumentar la resolución de entrada no se traduce aquí en una mejora de exactitud: a partir de 384 píxeles, el desempeño decrece de forma gradual, mientras que el tiempo de entrenamiento por época aumenta de manera notoria (de 28.0 s a 384 píxeles a 35.3 s a 512 píxeles y hasta 59.2 s a 768 píxeles). Más allá de la exactitud y el tiempo de entrenamiento, la resolución de entrada también incide de forma directa sobre el consumo de memoria del modelo, ya que el tamaño de los mapas de activación internos crece de forma proporcional al número de píxeles de la imagen; así, una imagen de 768 × 768 píxeles contiene cuatro veces más píxeles que una de 384 × 384, lo que se traduce en un mayor uso de memoria de la GPU durante el entrenamiento y la inferencia, incluso si no se modifica el tamaño de lote; este efecto es coherente con el incremento observado en el tiempo por época (de 28.0 s a 384 píxeles a 59.2 s a 768 píxeles), ya que un mayor consumo de memoria obliga a procesar más datos por paso y ralentiza cada iteración. Por este motivo, se adoptó la resolución de 384 × 384 píxeles para el modelo final: ofrece la mejor exactitud de las cuatro evaluadas, un costo de entrenamiento moderado (pues solo la resolución de 256 × 256 píxeles es más rápida, con 24.8 s por época) y un uso de memoria notablemente menor que las resoluciones de 512 y 768 píxeles, lo que además reduce el riesgo de agotar la memoria disponible en el servicio de inferencia en producción.

4.2.4 Evaluación cuantitativa del modelo seleccionado

El tercer eje de la evaluación se centró en cuantificar el desempeño global del modelo que finalmente se seleccionó, es decir, el que utiliza el codificador MiT-B0 con la configuración completa, con el fin de obtener una medida clara de qué tan bien resuelve la tarea de segmentar las enfermedades en las imágenes de maíz. Para esto, se calcularon sobre los conjuntos de validación y prueba las métricas definidas en la sección 3.5.1, empleando los pesos obtenidos con la media móvil exponencial (EMA)

de la época 29, y en la Tabla 56 se resumen los valores globales alcanzados por el modelo en ambos conjuntos.

Tabla 56. *Métricas globales del modelo sobre los conjuntos de validación y prueba.*

Métrica	Validación	Prueba
mIoU	0.8182	0.7593
Exactitud a nivel de píxel (Pixel Accuracy)	0.9767	0.9736
F1 macro	0.8972	0.8560
mAP	0.9468	0.9008

Fuente: Elaboración propia.

Como se aprecia en la Tabla 56, el modelo alcanzó un mIoU de 0.8182 en validación y de 0.7593 en prueba, con una exactitud a nivel de píxel de 0.9767 y 0.9736, respectivamente, lo que indica que la gran mayoría de los píxeles de cada imagen se clasificaron correctamente, incluso en datos completamente independientes del entrenamiento. Conviene aclarar que la exactitud a nivel de píxel resulta tan alta en buena medida porque la clase Fondo ocupa la mayor parte de la superficie de las imágenes y es la más fácil de segmentar, de modo que el mIoU y el F1 macro, que promedian por igual el desempeño de todas las clases, ofrecen una visión más realista del comportamiento del modelo frente a las clases de enfermedad. En ese sentido, un F1 macro de 0.8972 en validación (0.8560 en prueba) y un mAP de 0.9468 en validación (0.9008 en prueba) reflejan que el modelo mantiene un buen equilibrio entre precisión y exhaustividad incluso en las clases minoritarias, con una disminución moderada y consistente al pasar de validación a prueba, lo cual es un resultado sólido si se considera que se trata de segmentar cuatro clases sobre imágenes tomadas en campo, con condiciones reales de iluminación y de fondo.

Estos valores globales ocultan diferencias importantes entre una clase y otra, ya que no todas las enfermedades se segmentan con la misma facilidad: mientras la clase Fondo alcanza un IoU cercano a 0.97, las clases de enfermedad se sitúan aproximadamente entre 0.70 y 0.80 en validación, con caídas más marcadas en el conjunto de prueba, en particular en la clase Roya. Como complemento a las métricas globales y por clase, se generó la matriz de confusión del modelo final con los pesos EMA y con inferencia mediante TTA sobre los conjuntos de validación y de prueba, la cual permite observar de manera directa hacia dónde se dirigen los errores del modelo cuando confunde una clase con otra. Para cada conjunto se elaboró una figura independiente, la Figura 29 para validación y la Figura 30 para prueba, y en cada una se presentan dos versiones de esta misma matriz: la de valores absolutos,

expresados en número de píxeles, y la normalizada por fila, que equivale al recall de cada clase. A continuación se analiza cada conjunto por separado.

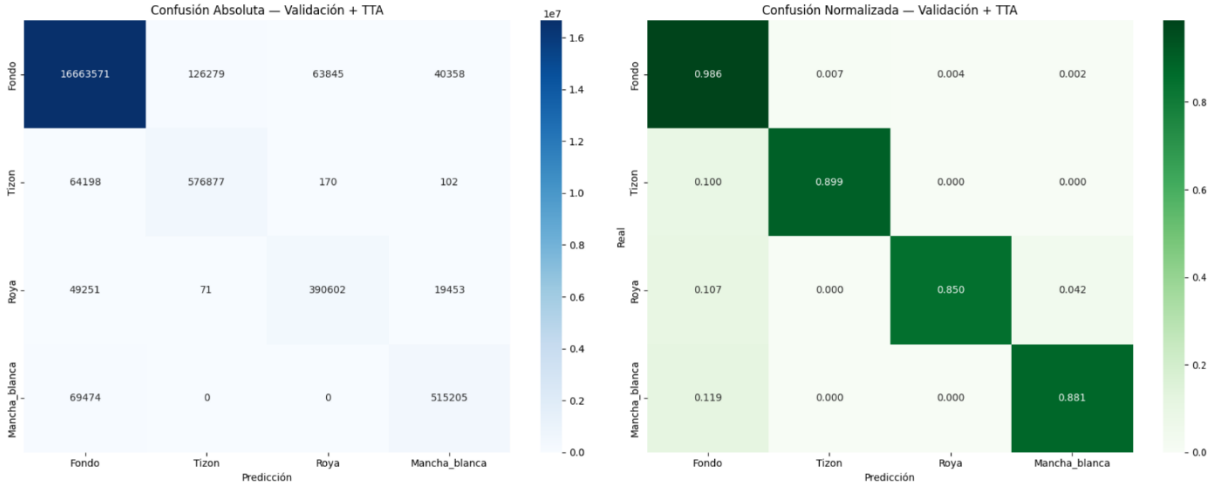


Figura 29. *Matriz de confusión normalizada por fila en validación.*

Fuente: Elaboración propia.

En la matriz de valores absolutos del conjunto de validación (Figura 29, izquierda) se aprecia que la clase Fondo concentra, como era de esperarse, la mayor cantidad de píxeles, con más de 16.6 millones clasificados de forma correcta, mientras que los errores más frecuentes se dan entre las clases de enfermedad y el fondo y no entre una enfermedad y otra, ya que, por ejemplo, del tizón se confunden 64 198 píxeles con fondo frente a apenas 170 con roya y 102 con mancha blanca, la roya aporta 49 251 píxeles mal clasificados como fondo y 19 453 como mancha blanca, y la mancha blanca suma 69 474 píxeles confundidos con fondo sin registrar confusión alguna con las otras dos enfermedades, lo que sugiere que el modelo distingue con bastante claridad entre los distintos tipos de lesión y que su principal fuente de error se concentra en los bordes de las lesiones, donde resulta más difícil separar el tejido enfermo del tejido sano que lo rodea.

La matriz normalizada por fila de validación (Figura 29, derecha), que en la práctica equivale al recall de cada clase, confirma esta lectura, ya que el 98.6 % de los píxeles reales de fondo se predicen como fondo, el 89.9 % de los del tizón se recuperan de forma correcta y el 88.1 % de los de mancha blanca también, mientras que la roya es la clase con el recall más bajo, con un 85.0 %, algo coherente con el IoU de 0.7463 que se reportó para esta clase en la Tabla 57; además, se observa que entre el 10.0 % y el 11.9 % de los píxeles de cada enfermedad terminan clasificados como fondo, frente a un porcentaje mucho menor de píxeles de fondo que se confunden con alguna enfermedad, entre el 0.24 % y el 0.75 %, de modo que el modelo tiende más bien a pasar por alto zonas afectadas pequeñas que a generar falsos positivos sobre tejido sano, un comportamiento razonable si se considera que las lesiones

de roya y de mancha blanca suelen ser de tamaño reducido y de bordes difusos, tal como ya se había señalado en la sección 4.2.6.

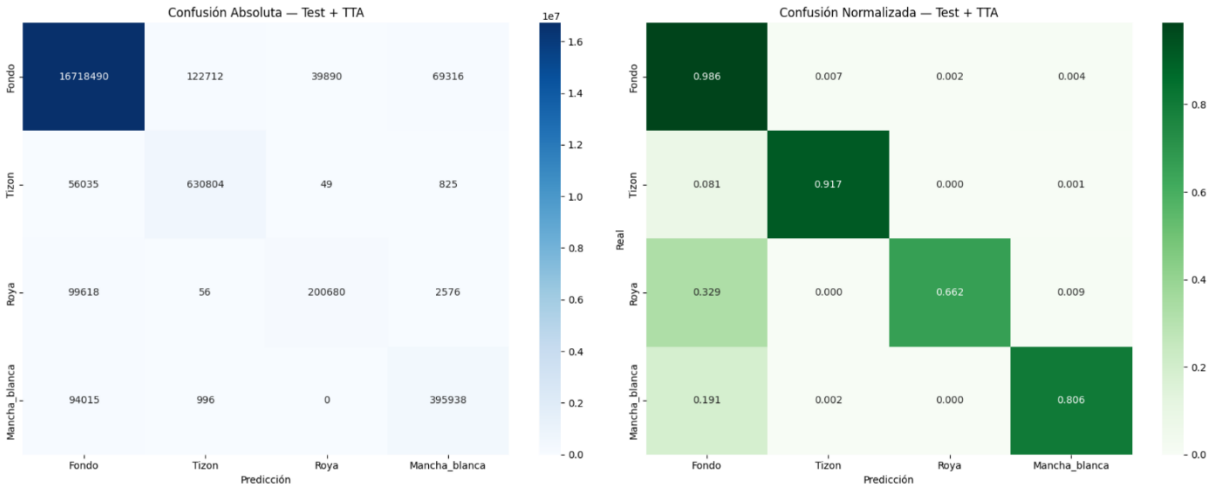


Figura 30. *Matriz de confusión del modelo seleccionado (MiT-B0) sobre el conjunto de prueba: valores absolutos y normalizados por fila.*

Fuente: Elaboración propia.

En la matriz de valores absolutos del conjunto de prueba (Figura 30, izquierda), la clase Fondo vuelve a concentrar la mayor cantidad de píxeles, con más de 16.7 millones clasificados de forma correcta, y los errores continúan concentrándose entre las clases de enfermedad y el fondo y no entre una enfermedad y otra: del tizón se confunden 56 035 píxeles con fondo frente a apenas 49 con roya y 825 con mancha blanca, la roya aporta 99 618 píxeles mal clasificados como fondo, 56 como tizón y 2 576 como mancha blanca, y la mancha blanca suma 94 015 píxeles confundidos con fondo y 996 con tizón, sin registrar confusión alguna con la roya. Cabe destacar que, en este conjunto, los 99 618 píxeles de roya omitidos representan una proporción mucho mayor frente a los 200 680 que sí se segmentaron correctamente que la observada en validación, donde se omitieron 49 251 píxeles frente a 390 602 acertados, lo que anticipa la caída de desempeño de esta clase que se detalla a continuación con la matriz normalizada.

La matriz normalizada por fila del conjunto de prueba (Figura 30, derecha) confirma esta lectura, ya que el fondo y el tizón conservan un comportamiento similar al de validación, con recalls de 98.6 % y 91.7 %, respectivamente, pero la roya cae a un recall de 66.2 %, ya que 99 618 de sus 302 930 píxeles reales (32.9 %) se clasifican como fondo, y la mancha blanca desciende a 80.6 %, con 94 015 píxeles (19.1 %) confundidos con fondo. Esto coincide con el IoU de 0.5853 y el recall de 0.6625 que se reportan para la roya en la sección 4.2.6 y confirma que el principal error del modelo ante datos nuevos es omitir regiones afectadas de roya y de mancha blanca, más que confundir unas enfermedades con otras, ya que las confusiones entre enfermedades se mantienen por debajo del 1 % de los píxeles de

cada clase (la mayor, de roya clasificada como mancha blanca, es del 0.9 %). En comparación con la validación, donde la roya alcanzaba un recall de 85.0 %, la caída de cerca de 19 puntos porcentuales indica una menor capacidad de generalización en esta clase; además, entre el 8.1 % y el 32.9 % de los píxeles de cada enfermedad terminan asignados al fondo (frente al 10.0 % y el 11.9 % en validación), mientras que solo entre el 0.24 % y el 0.72 % de los píxeles de fondo se predicen como alguna enfermedad, de modo que, al igual que en validación, el error dominante sigue siendo la omisión de zonas afectadas y no la generación de falsos positivos sobre tejido sano.

4.2.5 Análisis del proceso de aprendizaje

El cuarto eje de la evaluación se centró en analizar cómo aprendió el modelo a lo largo del entrenamiento, ya que observar la evolución de la pérdida y del mIoU época tras época permite entender si el modelo aprendió de forma estable o si tuvo problemas como el sobreajuste. Para ello se registraron, en cada época, la pérdida sobre el conjunto de entrenamiento, la pérdida sobre el conjunto de validación y el mIoU de validación, y en la Figura 31 se muestran las curvas correspondientes al modelo final con el codificador MiT-B0 a 384 píxeles.

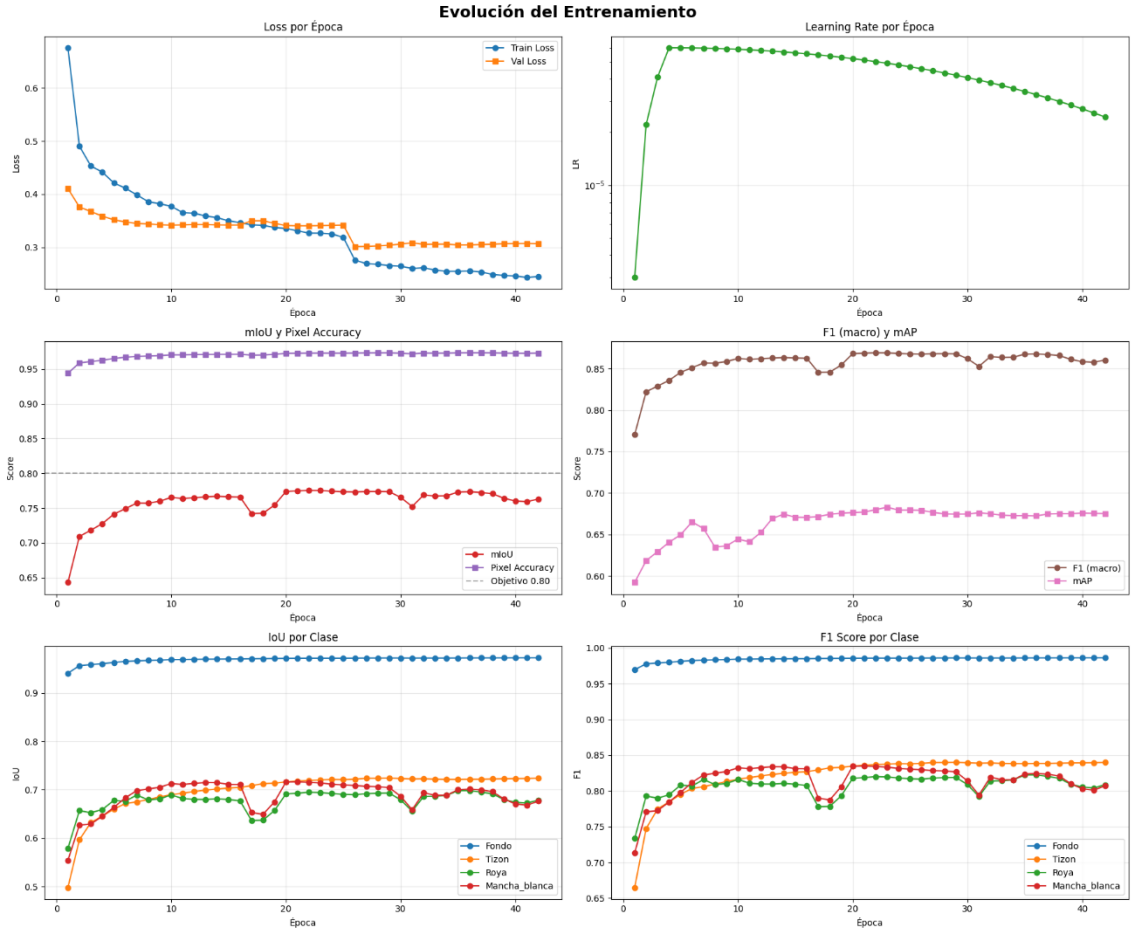


Figura 31. *Curvas de pérdida de entrenamiento y validación y evolución del modelo MiT-B0.*

Fuente: Elaboración propia.

En las curvas se observa una fase inicial de crecimiento pronunciado tanto en el mIoU como en la exactitud a nivel de píxel (Píxel Accuracy), seguida de una estabilización progresiva conforme avanza el entrenamiento. La Píxel Accuracy asciende con mayor rapidez y se estabiliza en niveles altos desde las primeras épocas, un comportamiento consistente con el predominio espacial de la clase Fondo, cuyo IoU supera 0.97 en el conjunto de validación. El mIoU, en cambio, exhibe un ascenso más gradual, ya que promedia el solapamiento espacial de las cuatro clases por igual y es más sensible a los errores de contorno en las clases minoritarias (Tizón, Roya y Mancha Blanca), por lo que continúa mejorando durante más épocas antes de estabilizarse cerca de su valor final de 0.8182. El F1 macro y el mAP siguen un patrón similar: el F1 macro traza una trayectoria intermedia entre las curvas de mIoU y Píxel Accuracy, reflejando el equilibrio entre el desempeño sobresaliente en Fondo y el más moderado en las clases de enfermedad, mientras que el mAP se estabiliza con mayor rapidez y alcanza el valor más alto del conjunto (0.9468).

En conjunto, ambas gráficas muestran una fase inicial de crecimiento pronunciado seguida de una estabilización progresiva de todas las curvas, sin evidencia de degradación posterior que sugiera sobreajuste. Este comportamiento es consistente con la convergencia del modelo mediante la estrategia de parada temprana, que detuvo el entrenamiento en la época 49 de las 70 planificadas al no haber mejora del mIoU durante 20 épocas seguidas, de modo que el modelo final no corresponde a la última época sino a los pesos de la media móvil exponencial (EMA) guardados en la época 29, que fueron los de mejor desempeño en validación. El entrenamiento completo tomó 22.90 minutos, a un promedio de 28.0 segundos por época. De esta forma, las curvas confirman que el entrenamiento fue estable y que la combinación de la parada temprana y la EMA evitó que el sobreajuste afectara al modelo final.

4.2.6 Evaluación por categoría de enfermedad

Una vez analizado el desempeño global del modelo, resulta necesario profundizar en el detalle de cada categoría, ya que el promedio general puede ocultar comportamientos muy distintos entre una enfermedad y otra. Por este motivo, en esta sección se examina el desempeño del modelo MiT-B0 para cada una de las cuatro clases por separado, considerando el IoU, la precisión, el recall, el F1 y la precisión promedio (AP), cuyos valores se resumen en la Tabla 57 para el conjunto de validación y en la Tabla 58 para el conjunto de prueba.

Tabla 57. *Métricas de segmentación por clase sobre el conjunto de validación.*

Clase	IoU	Precisión	Recall	F1	AP
Fondo	0.9758	0.9891	0.9864	0.9877	0.9978
Tizón	0.7514	0.8203	0.8995	0.8581	0.9340
Roya	0.7463	0.8592	0.8503	0.8547	0.9223
Mancha Blanca	0.7993	0.8958	0.8812	0.8884	0.9329

Fuente: Elaboración propia.

Como era de esperar, la clase Fondo es la que mejor se segmenta, con un IoU de 0.9758 y un F1 de 0.9877, lo cual se explica porque es la clase mayoritaria y presenta límites más definidos frente a las lesiones. En cuanto a las enfermedades, sobre el conjunto de validación la mancha blanca obtiene el mejor IoU (0.7993) y F1 (0.8884), seguida de tizón (IoU 0.7514) y roya (IoU 0.7463), mientras que el tizón alcanza el mayor recall (0.8995) y el AP más elevado (0.9340), lo que indica que el modelo logra recuperar la mayoría de las regiones afectadas por esta enfermedad, algo coherente con el hecho de que produce lesiones alargadas y bien contrastadas que resultan más fáciles de delimitar.

En validación, la roya presenta el desempeño algo menor entre las tres enfermedades, con el IoU más bajo (0.7463) y un AP de 0.9223, aunque mantiene una precisión de 0.8592, lo que se relaciona

con que sus lesiones suelen ser pequeñas, dispersas y de tonos que se confunden con facilidad con el fondo o con las etapas iniciales de otras enfermedades. Aun así, los valores de AP se mantienen altos en las tres enfermedades, entre 0.9223 y 0.9340, lo que muestra que el modelo ordena bien sus predicciones según el nivel de confianza en el conjunto de validación. Sin embargo, esta lectura cambia de forma notable al evaluar sobre el conjunto de prueba, formado por imágenes completamente independientes del entrenamiento, como se detalla a continuación.

Tabla 58. *Métricas de segmentación por clase sobre el conjunto de prueba.*

Clase	IoU	Precisión	Recall	F1
Fondo	0.9720	0.9853	0.9863	0.9858
Tizón	0.7774	0.8360	0.9172	0.8747
Roya	0.5853	0.8340	0.6625	0.7384
Mancha Blanca	0.7024	0.8448	0.8065	0.8252

Fuente: Elaboración propia.

Sobre el conjunto de prueba (Tabla 58), la clase Fondo conserva el mejor desempeño (IoU 0.9720, F1 0.9858), coherente con su predominio espacial y su patrón visual homogéneo. El tizón muestra un comportamiento sólido (IoU 0.7774, F1 0.8747), incluso por encima de su resultado en validación, mientras que la mancha blanca se mantiene en un nivel intermedio (IoU 0.7024, F1 0.8252). La roya, en cambio, presenta la mayor caída de desempeño respecto a validación, con un IoU de apenas 0.5853 y un recall de 0.6625, lo que indica una menor capacidad del modelo para detectar todas las regiones afectadas por esta enfermedad en imágenes no vistas; esta diferencia sugiere que la roya es la clase más sensible a la variabilidad del conjunto de prueba, probablemente por la menor cantidad de ejemplos de entrenamiento disponibles para esta categoría. En conjunto, estos resultados confirman que el modelo mantiene un desempeño equilibrado entre las cuatro clases en validación, con un comportamiento sobresaliente en el fondo y buen desempeño en tizón y mancha blanca, pero revelan a la roya como la principal limitación del modelo frente a datos no vistos; el análisis visual de estos aciertos y errores se complementa más adelante con los mapas de interpretabilidad de la sección 4.2.7.

4.2.7 Interpretabilidad mediante Seg-Grad-CAM

Más allá de las métricas numéricas, en un sistema pensado para apoyar al agricultor, resulta fundamental entender en qué se fija el modelo al tomar una decisión, ya que un buen desempeño cuantitativo no sirve de mucho si el modelo acierta por las razones equivocadas. Por este motivo, se aplicó una técnica de inteligencia artificial explicable, en concreto, Seg-Grad-CAM, adaptación de Grad-CAM a las tareas de segmentación, que permite generar mapas de calor que resaltan las regiones de la imagen que más influyen en la predicción de cada clase, tal como se describió en la sección 3.4.7.

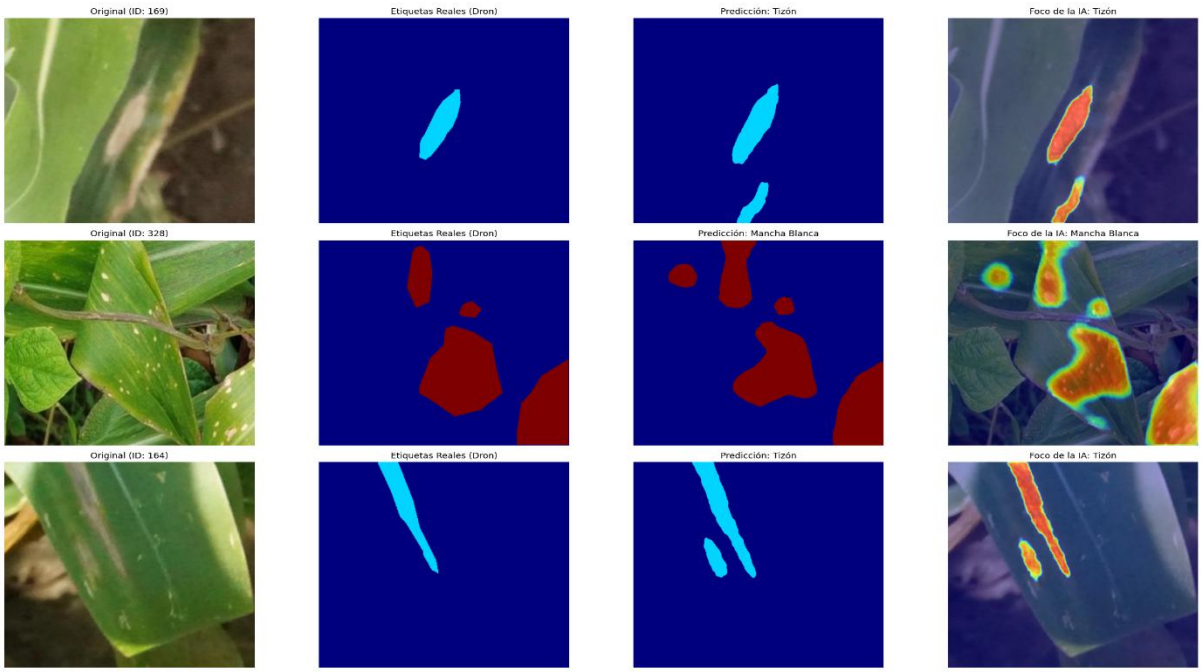


Figura 32. *Mapas de calor Seg-Grad-CAM*

Fuente: Elaboración propia.

En la Figura 32 se presentan varios ejemplos en los que se superpone el mapa de calor sobre la imagen original, de manera que las zonas en tonos más cálidos corresponden a las regiones que el modelo consideró más importantes para identificar cada enfermedad. Se observa que, en la mayoría de los casos, el modelo concentra su atención sobre las propias lesiones de la hoja y no sobre el fondo ni sobre las partes sanas de la planta, lo que indica que las predicciones se apoyan en las características visuales correctas de cada patología y no en elementos ajenos a la enfermedad.

Este comportamiento es coherente con los resultados cuantitativos, pues en el tizón, que fue la enfermedad mejor segmentada, los mapas resaltan de forma clara y continua las lesiones alargadas, mientras que en la roya y en la mancha blanca la atención aparece más dispersa y en ocasiones se reparte entre varias manchas pequeñas, lo que ayuda a explicar por qué esas dos clases resultan más difíciles de delimitar. En conjunto, estos mapas aportan una capa de confianza al sistema, ya que permiten al usuario y a un eventual experto verificar que el modelo está mirando la parte enferma de la hoja, y refuerzan la utilidad de la herramienta como un apoyo al diagnóstico y no como una caja negra.

4.2.8 Eficiencia computacional y viabilidad de despliegue

El último eje de la evaluación se centró en la eficiencia computacional del modelo, ya que uno de los objetivos de este trabajo era obtener un modelo liviano que pudiera integrarse en una aplicación web

y responder en un tiempo razonable. Por esta razón, además del desempeño de segmentación, se midieron el número de parámetros, el tamaño del modelo en disco y la velocidad de inferencia, cuyos valores se resumen en la Tabla 59.

Tabla 59. *Métricas de eficiencia computacional del modelo seleccionado (MiT-B0).*

Métrica	Valor
Parámetros	4 136 820
Tamaño del modelo	15.78 MB
Latencia de inferencia (GPU, sin TTA)	14.20 ms/imagen
Velocidad de inferencia (GPU, sin TTA)	70.4 FPS
Latencia de inferencia (GPU, con TTA)	98.63 ms/imagen
Velocidad de inferencia (GPU, con TTA)	10.1 FPS
Latencia de inferencia en producción (CPU, segmentación)	115.7 ms/imagen (media); 113.2 ms (mediana)
Tiempo total de entrenamiento	22.90 min
Tiempo promedio por época	28.0 s

Fuente: Elaboración propia. Las mediciones en GPU se realizaron sin TTA y con TTA; la latencia en producción corresponde al benchmark de la Tabla 65 (servidor sobre CPU).

Como se aprecia en la Tabla 59, el modelo final es sumamente liviano, ya que cuenta con 4 136 820 parámetros y ocupa apenas 15.78 MB en disco, un tamaño muy reducido si se lo compara con las variantes más pesadas evaluadas en la sección 4.2.2, como MiT-B1 (54.57 MB) o MiT-B2 (96.70 MB), que son entre tres y seis veces más grandes. En cuanto a la velocidad, en GPU y sin TTA, el modelo procesa cada imagen en 14.20 milisegundos (70.4 imágenes por segundo), mientras que con TTA la latencia sube a 98.63 milisegundos (10.1 imágenes por segundo); por su parte, en el servidor de producción, que se ejecuta en CPU, la segmentación de una imagen toma en promedio 115.7 milisegundos (mediana de 113.2 ms), medición que se detalla en la sección 4.3.5. En todos los casos los tiempos son suficientemente bajos para un uso interactivo y permiten entregar el diagnóstico de forma prácticamente inmediata.

Estas características hacen que el modelo sea perfectamente viable para su despliegue en la aplicación web, pues su tamaño reducido facilita cargarlo en el backend sin consumir demasiados recursos y su alta velocidad de inferencia asegura que el usuario reciba la respuesta sin esperas notorias. Cabe recordar que, tal como se mostró en el análisis de sensibilidad a la resolución de la sección 4.2.3, la resolución de 384 × 384 píxeles ofrece la mejor exactitud con un costo de entrenamiento moderado, pues solo la de 256 × 256 píxeles es más rápida (24.8 s por época frente a 28.0 s), de modo que para el despliegue en producción se conservó esta configuración y se prescindió de resoluciones mayores que

solo incrementarían el costo de inferencia sin mejorar el desempeño, aspecto que se retoma en los resultados de la aplicación web de la sección 4.3.

4.2.9 Comparación con el estado del arte

Para cerrar la evaluación del modelo, resulta útil situar los resultados obtenidos frente a los de otros trabajos del estado del arte que también abordan la detección de enfermedades del maíz a partir de imágenes capturadas con drones. Conviene aclarar desde el inicio que esta comparación es de carácter referencial y no una comparación directa, ya que cada trabajo emplea su propio conjunto de datos, se enfoca por lo general en una sola enfermedad y utiliza métricas que no siempre coinciden, mientras que el presente trabajo segmenta de forma simultánea tres enfermedades (tizón, roya y mancha blanca) además del fondo. En la Tabla 60 se resumen los principales trabajos considerados junto con su enfoque y los resultados que reportan.

Tabla 60. *Comparación referencial del modelo propuesto frente a trabajos del estado del arte en segmentación de enfermedades del maíz con imágenes de dron.*

Trabajo	Año	Modelo	Cultivo	N° clases	Enfermedades	mAP	mIoU	F1	N° parámetros
Chen et al. [37]	2023	VGG16 + ABB + FRM	Maíz	1	Tizón del norte (NLB)	N/A	0.508	0.674	N/A
Soomro et al. [40]	2026	SilvaFormer (MiT + SegFormer)	Pino (forestal)	1	Pine Wilt Disease	N/A	0.7701	0.8474 (Dice)	6.11 M
Yang et al. [41]	2024	SE-Swin UNet	Maíz	1	Large Spot, Small Spot, Stripe Rust	N/A	0.8461 (prueba)	0.8851	122.58 M
Wen et al. [42]	2025	YOLOv5-Mobile-Seg	Maíz	1	Tizón del norte (NCLB)	0.888	N/A	N/A	4.66 M
Zhang S. et al. [43]	2026	TDAVM-UNet	Multicultivo	1	Regiones enfermas	N/A	0.8222	0.8381	26.87 M
Este trabajo	2026	DeepLabV3 + con MiT-B0 (liviano)	Maíz	4	Tizón, Roya, Mancha Blanca y Fondo	0.9468	0.8182	0.8972	4.136 M

Fuente: Elaboración propia.

Como se observa en la Tabla 60, el modelo propuesto se ubica en un nivel competitivo dentro de este grupo de trabajos. El estudio de Chen et al. [37], que emplea un enfoque débilmente supervisado basado en VGG16 con los módulos ABB y FRM para segmentar el tizón del norte, reporta un mIoU de 0.508 y un F1 de 0.674 sobre una sola enfermedad, valores inferiores al mIoU global de 0.8182 y al F1 de 0.8972 que aquí se alcanzan considerando las cuatro clases a la vez. SilvaFormer (Soomro et al., 2026), que combina MiT y SegFormer para segmentar la Pine Wilt Disease en imágenes UAV, alcanza

un mIoU de 0.7701 y un F1 (Dice) de 0.8474 con apenas 6.11 millones de parámetros, cifras cercanas a las del presente modelo pero obtenidas sobre una tarea de una sola clase y en un cultivo distinto (pino forestal). Por su parte, SE-Swin UNet (Yang et al., 2024), que combina Swin Transformer y U-Net para segmentar tres tipos de lesiones en maíz (Large Spot, Small Spot y Stripe Rust), reporta un mIoU de 0.8461 y un F1 de 0.8851 sobre el conjunto de prueba, aunque con un costo de 122.58 millones de parámetros, casi 30 veces más que el modelo aquí propuesto.

El trabajo más cercano en su filosofía al presente es el de Wen et al. (2025), ya que también busca un modelo liviano para la segmentación del tizón del norte mediante YOLOv5-Mobile-Seg; ese modelo alcanza una precisión promedio (AP) del 88.8 % utilizando 4.66 millones de parámetros, mientras que el modelo aquí propuesto obtiene un mAP de 0.9468 sobre tres enfermedades, lo que resulta muy favorable si se considera que enfrenta un problema multiclase más complejo. TDAVM-UNet (Zhang S. et al., 2026), que incorpora mecanismos de atención guiados por la tarea sobre una arquitectura VM-UNet para segmentar regiones enfermas en múltiples cultivos, obtiene un mIoU de 0.8222 y un F1 de 0.8381 con 26.87 millones de parámetros. En conjunto, esta comparación sugiere que la combinación de DeepLabV3+ con el codificador MiT-B0 ofrece un desempeño acorde con el estado del arte, con la ventaja adicional de segmentar varias enfermedades a la vez, mantener un tamaño reducido de 15.78 MB (4 136 820 parámetros) e incorporar interpretabilidad mediante Seg-Grad-CAM. No obstante, se insiste en que estas cifras deben leerse con cautela, pues provienen de conjuntos de datos, cultivos y tareas diferentes, aspecto que se retoma dentro de las limitaciones del estudio en la sección 4.2.10.

4.2.10 Discusión general y limitaciones

A lo largo de esta sección se evaluó de forma integral el modelo de segmentación desarrollado para la detección de enfermedades en el maíz, y los resultados obtenidos permiten afirmar que el objetivo de contar con un modelo preciso y a la vez liviano se cumplió de manera satisfactoria. El modelo final, basado en la arquitectura DeepLabV3+ con el codificador MiT-B0 a 384 píxeles, alcanzó un mIoU de 0.8182, un F1 macro de 0.8972 y un mAP de 0.9468 sobre el conjunto de validación (0.7593 de mIoU sobre el conjunto de prueba), segmentando de forma simultánea el fondo y las tres enfermedades de interés, y todo ello ocupando apenas 15.78 MB y con una velocidad de inferencia de 70.4 imágenes por segundo. El análisis por clase mostró que el fondo y el tizón son los que mejor se segmentan, mientras que la roya y la mancha blanca concentran el mayor margen de mejora, y los mapas de interpretabilidad confirmaron que el modelo apoya sus decisiones en las lesiones de la hoja y no en elementos ajenos a la enfermedad.

Un aspecto que merece una discusión especial es el resultado del análisis de sensibilidad a la resolución de entrada (sección 4.2.3), que mostró que aumentar la resolución más allá de 384 píxeles no se traduce en una mejora de exactitud, sino en un incremento notorio del tiempo de entrenamiento.

Este comportamiento es coherente con la comparación frente al estado del arte, donde el modelo se ubicó en un nivel competitivo respecto a trabajos que abordan la misma enfermedad, con la ventaja de segmentar varias enfermedades a la vez y de mantenerse liviano, aunque siempre teniendo presente que esos trabajos emplean conjuntos de datos distintos.

No obstante, es necesario reconocer con honestidad las limitaciones de este estudio, agrupadas en cinco ámbitos. En cuanto al dataset, el conjunto de datos sigue siendo limitado en tamaño y proviene mayoritariamente de una zona geográfica específica (Pogllocunga y alrededores de Otavalo), lo que puede condicionar el desempeño del modelo en cultivos con condiciones distintas de iluminación, variedad o estado fenológico, y el uso de datasets públicos complementarios para la clase Tizón introduce cierta heterogeneidad en las condiciones de captura respecto a las imágenes propias. En cuanto al modelo, aunque el desempeño global fue sólido y se confirmó sobre un conjunto de prueba independiente, la clase Roya mostró una caída notable de desempeño entre validación y prueba (IoU de 0.7463 a 0.5853), la mayor de las cuatro clases, lo que sugiere que el modelo generaliza menos bien para esta enfermedad, posiblemente por la menor cantidad de ejemplos de entrenamiento disponibles para esta categoría; junto con la mancha blanca, las clases visualmente complejas, con lesiones pequeñas y de bordes difusos, concentraron el mayor margen de error. En términos de generalización, estos resultados indican que el modelo generaliza de forma adecuada para el fondo y el tizón, cuyas métricas se mantienen estables o incluso mejoran entre validación y prueba, mientras que para la roya la generalización es claramente más limitada: la caída de desempeño en datos no vistos sugiere que el conjunto de validación, por sí solo, sobreestima la fiabilidad del modelo para esta clase. En términos prácticos, esto implica que, en un escenario de uso real, las predicciones del modelo para la roya deberían interpretarse con mayor cautela y, de ser posible, complementarse con una revisión humana, mientras que las predicciones para el tizón y el fondo pueden considerarse más confiables. En cuanto a la captura, las imágenes se obtuvieron únicamente en el espectro RGB mediante un UAV, sin sensores multiespectrales, y las condiciones de iluminación y la altura de vuelo durante las jornadas de captura pudieron introducir variaciones que afectan la consistencia visual de las lesiones entre imágenes. En cuanto a las limitaciones experimentales, cada configuración se entrenó con una sola semilla de aleatorización sin repeticiones que permitan estimar la variabilidad de los resultados, lo que impide reportar intervalos de confianza sobre las métricas obtenidas. Finalmente, en cuanto al software, tanto el entrenamiento como el despliegue dependen de servicios en la nube (Google Colab, Hugging Face Spaces, Render, Supabase) y por tanto de una conexión a Internet estable, lo que puede limitar su uso en zonas rurales con conectividad restringida o infraestructura limitada.

A partir de estas limitaciones se abren varias líneas de trabajo futuro, entre las que destacan la ampliación del conjunto de prueba con imágenes capturadas en otras zonas y campañas, la repetición de los entrenamientos con varias semillas para reportar intervalos de confianza, y la ampliación del

conjunto de datos con más ejemplos de roya y de mancha blanca, que fueron las clases más difíciles. Con todo, los resultados alcanzados respaldan la viabilidad del modelo como componente central de la aplicación web de diagnóstico, cuyo desempeño con usuarios reales se aborda en la siguiente sección.

4.3 Resultados de la aplicación web

El proyecto se materializó en una aplicación web funcional que integra el modelo de segmentación entrenado, cuya eficiencia computacional se detalló en la sección 4.2.8 (Tabla 59) y lo pone a disposición del usuario final para el diagnóstico de enfermedades en cultivos de maíz a partir de imágenes capturadas con dron. De esta forma, la aplicación quedó organizada en torno a tres módulos principales: autenticación de usuarios, análisis y diagnóstico de imágenes, e historial de resultados. En la Tabla 61 se resumen las tecnologías empleadas en cada capa de la aplicación.

Tabla 61. *Tecnologías utilizadas en la aplicación web.*

Capa	Tecnología	Función
Frontend	React	Interfaz de usuario: carga de imágenes y visualización de resultados
Backend	FastAPI (Python)	API REST, carga del modelo e inferencia de segmentación
Modelo	DeepLabV3+ (encoder MiT-B0, PyTorch)	Segmentación semántica de enfermedades y cálculo del área afectada
Explicabilidad	Seg-Grad-CAM	Generación de los mapas de calor de las predicciones
Base de datos	PostgreSQL	Almacenamiento de usuarios e historial de análisis

Fuente: Elaboración propia.

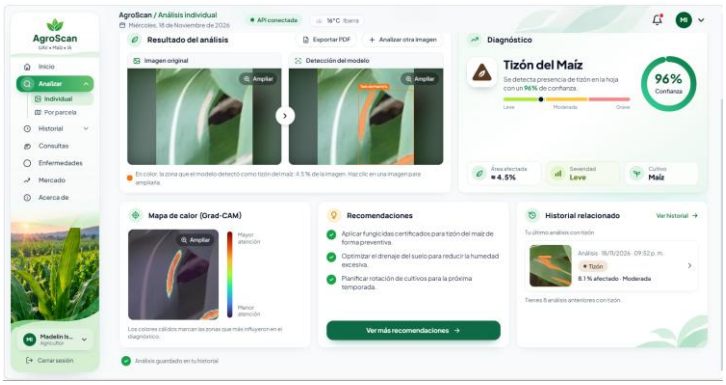


Figura 33. *Interfaz de la aplicación web con el resultado de un análisis.*

Fuente: Elaboración propia.

En conjunto, y como se aprecia en la Figura 33, la aplicación web desarrollada cumple con el objetivo de llevar el modelo entrenado a un entorno accesible y fácil de usar, permitiendo que un usuario sin conocimientos técnicos obtenga, en pocos pasos, un diagnóstico visual de las enfermedades presentes en su cultivo a partir de una simple imagen capturada por un dron.

4.3.1 Diseño del instrumento de evaluación de la aplicación

El instrumento que se usó para evaluar la eficiencia y la usabilidad que perciben los usuarios se armó como un cuestionario estructurado y se implementó en Microsoft Forms, con el fin de recoger la opinión de los participantes después de que usaran la aplicación web, y se organizó en cuatro secciones que van de la mano con las características de calidad en uso que se consideran en este estudio, es decir, una primera sección para caracterizar al encuestado, dos secciones centradas en la usabilidad y en la eficiencia de desempeño, y una última sección de valoración global y comentarios abiertos.

La Sección A, Datos generales del encuestado, busca caracterizar el perfil de los participantes y está compuesta por cuatro preguntas cerradas, donde la primera es el cargo o rol que desempeña, es decir, agricultor o productor agrícola, ingeniero agrónomo, técnico agrícola, estudiante del área agropecuaria, técnico de una institución pública o privada relacionada al sector agrícola, u otra; la segunda son los años de experiencia en el área agrícola, ya sea menos de 1 año, entre 1 y 3 años, entre 4 y 7 años, o más de 7 años; la tercera es el nivel de experiencia usando aplicaciones o dispositivos tecnológicos, que puede ser ninguna, básica, intermedia o avanzada; y la cuarta pregunta si el participante ya usó antes alguna aplicación agrícola similar, respondiendo sí o no.

La Sección B, Usabilidad, agrupa trece enunciados que se valoran con una escala Likert de cinco niveles, la cual se detalla en el siguiente apartado, y abordan las subcaracterísticas de usabilidad que define la norma ISO/IEC 25010, es decir, la capacidad para reconocer su idoneidad, que revisa qué tanto

se entiende el propósito de la aplicación y las funciones que ofrece; la capacidad de aprendizaje, que mide qué tan fácil es aprender a usarla y completar el proceso de carga y diagnóstico; la operabilidad, que revisa la navegación entre secciones, la gestión de cuenta y el número de pasos para obtener un resultado; la protección contra errores de usuario, que revisa la claridad de las advertencias y la posibilidad de corregir errores sin perder la información; la estética de la interfaz, que revisa el diseño visual y la opción de tener un tema claro u oscuro; y la accesibilidad, que revisa el tamaño y el contraste de textos, botones e íconos, además del uso de la aplicación desde el navegador sin instalar nada adicional.

La Sección C, Eficiencia de desempeño, incluye siete enunciados que se evalúan con la misma escala Likert y que se refieren a dos subcaracterísticas, por un lado el comportamiento temporal, es decir, el tiempo de carga de la aplicación y el tiempo en que se genera el diagnóstico y los mapas de calor, y por otro lado la utilización de recursos y la capacidad, que abarca la fluidez de funcionamiento, la ausencia de bloqueos o cierres inesperados, la respuesta correcta ante imágenes de distinto tamaño o varios análisis seguidos, y el correcto almacenamiento y recuperación del historial de análisis.

Finalmente, la Sección D, Valoración global y comentarios, cierra el instrumento con una pregunta cerrada sobre la disposición del participante a recomendar la aplicación a otros agricultores o técnicos agrícolas, respondiendo sí o no, y con dos preguntas abiertas orientadas a identificar los aspectos mejor valorados y las dificultades encontradas o las posibles mejoras, de modo que, en conjunto, el instrumento reúne nueve preguntas estructuradas, de las cuales veinte enunciados se valoran mediante escala Likert repartidos entre usabilidad y eficiencia de desempeño.

4.3.2 Escala de valoración y criterios de interpretación

La escala que se empleó en las secciones de usabilidad y de eficiencia de desempeño del instrumento corresponde a una escala Likert de cinco niveles, es decir, Totalmente en desacuerdo, En desacuerdo, Ni de acuerdo ni en desacuerdo, De acuerdo y Totalmente de acuerdo, y a cada nivel se le asignó un valor ordinal de 1 a 5 respectivamente, lo que permite calcular la media de las respuestas que se obtienen para cada enunciado y para cada indicador evaluado.

Para interpretar los resultados, el rango teórico de la escala, que va de 1 a 5, se dividió en cinco intervalos de igual amplitud de 0.80, obteniendo así los niveles de valoración que se muestran en la Tabla 62:

Tabla 62. *Criterios de interpretación de la escala de valoración.*

Rango de la media	Nivel de valoración
1.00 – 1.80	Muy bajo
1.81 – 2.60	Bajo
2.61 – 3.40	Medio
3.41 – 4.20	Alto
4.21 – 5.00	Muy alto

Fuente: Elaboración propia.

Esta escala de interpretación se aplicó de manera uniforme tanto a los enunciados de la Sección B, Usabilidad, como a los de la Sección C, Eficiencia de desempeño, de modo que la percepción promedio de los usuarios respecto a cada indicador evaluado se puede clasificar de forma consistente.

4.3.3 Correspondencia entre los indicadores evaluados y la norma ISO/IEC 25010

Con el fin de mantener coherencia con el modelo de calidad presentado en el apartado 2.5, los indicadores considerados en el instrumento se organizaron según las características y subcaracterísticas de la norma ISO/IEC 25010 que corresponden a la usabilidad y a la eficiencia de desempeño, las dos características de calidad que se validan en esta evaluación. En la Tabla 63 se presenta dicha correspondencia.

Tabla 63. *Correspondencia entre los indicadores del instrumento y la norma ISO/IEC 25010.*

Característica ISO/IEC 25010	Subcaracterística	Sección del instrumento
Usabilidad	Capacidad para reconocer su idoneidad	Sección B, ítems 1-2
Usabilidad	Capacidad de aprendizaje	Sección B, ítems 3-4
Usabilidad	Capacidad para ser usado (operabilidad)	Sección B, ítems 5-7
Usabilidad	Protección contra errores de usuario	Sección B, ítems 8-9
Usabilidad	Estética de la interfaz de usuario	Sección B, ítems 10-11
Usabilidad	Accesibilidad	Sección B, ítems 12-13
Eficiencia de desempeño	Comportamiento temporal	Sección C, ítems 1-3
Eficiencia de desempeño	Utilización de recursos y capacidad	Sección C, ítems 4-7

Fuente: Elaboración propia.

Esta correspondencia permite que los resultados presentados en los apartados 4.3.5, sobre eficiencia percibida, y 4.3.6, sobre usabilidad percibida, se interpreten directamente en términos de las

características de calidad definidas por la norma, evitando así ambigüedades en el análisis de la evaluación de la aplicación web. El tamaño de la muestra y el perfil de los participantes que respondieron el instrumento se detallan en el apartado 4.3.4, mientras que las limitaciones de generalización derivadas de dicho tamaño muestral se discuten en los apartados 4.3.6 y 4.3.8.

4.3.4 Perfil de los usuarios participantes en la evaluación

Hasta este corte de análisis, la aplicación web ha sido evaluada por 8 usuarios mediante el instrumento descrito en el apartado 4.3.1. Este número corresponde a las respuestas recogidas hasta la fecha, ya que la recolección sigue en curso, por lo que los resultados de los apartados 4.3.5 a 4.3.8 se irán actualizando conforme se sumen nuevas respuestas.

En cuanto al cargo o rol que desempeñan, la mayoría de los participantes fueron estudiantes del área agropecuaria, es decir, 4 de 8, seguidos de los agricultores o productores agrícolas con 2, y un participante en cada una de las categorías de técnico agrícola y agroindustrial, mientras que en los años de experiencia en el área agrícola 6 participantes señalaron tener entre 1 y 3 años, 1 indicó entre 4 y 7 años y 1 más de 7 años.

En cuanto a la experiencia usando aplicaciones o dispositivos tecnológicos, la mayoría se ubicó en un nivel intermedio con 4 participantes, seguido del nivel básico con 3 y de una persona sin experiencia previa, y por último el uso previo de una aplicación agrícola similar quedó dividido por igual, ya que 4 participantes indicaron que sí habían usado antes una herramienta parecida y 4 que no.

4.3.5 Valoración de la eficiencia percibida en el uso de la aplicación

Con base en las 8 respuestas que se recogieron hasta este corte, se calculó la media de cada uno de los siete enunciados que integran la Sección C del instrumento, es decir, la Eficiencia de desempeño, que corresponden a las subcaracterísticas de comportamiento temporal y de utilización de recursos y capacidad. Los resultados se presentan en la Tabla 64.

Tabla 64. *Valoración de la eficiencia percibida por enunciado.*

Enunciado	Media	Nivel
El tiempo de carga de la aplicación al abrirla en el navegador fue adecuado.	4.25	Muy alto
El diagnóstico de la imagen se generó en un tiempo razonable.	4.38	Muy alto
El mapa de calor y los resultados del análisis se mostraron sin demoras excesivas.	4.25	Muy alto
La aplicación funcionó de manera fluida, sin trabarse ni ponerse lenta.	4.38	Muy alto
No experimenté bloqueos, cierres inesperados ni congelamientos.	4.62	Muy alto
La aplicación respondió correctamente al cargar imágenes de distinto tamaño o al realizar varios análisis seguidos.	4.62	Muy alto
El historial almacenó y recuperó correctamente todos mis análisis previos.	4.50	Muy alto

Fuente: Elaboración propia, con base en las 8 respuestas recopiladas hasta la fecha.

La media global de eficiencia percibida, considerando los siete enunciados, fue de 4.43 sobre 5, lo que corresponde a un nivel de valoración muy alto según la escala de interpretación que se definió en el apartado 4.3.2, ya que los enunciados mejor valorados fueron la ausencia de bloqueos o cierres inesperados y la respuesta correcta ante imágenes de distinto tamaño o análisis sucesivos, ambos con una media de 4.62, mientras que el tiempo de carga de la aplicación y la generación de los mapas de calor obtuvieron las valoraciones más bajas del grupo con 4.25, aunque igualmente dentro del nivel Muy alto.

Además de la percepción de los usuarios, se midió de forma objetiva el rendimiento de la aplicación desplegada en producción, a partir del tiempo de carga (de la interfaz web y del modelo), el tiempo de análisis y el tiempo de respuesta de la API. Para ello se emplearon las 125 imágenes del conjunto de prueba, que se enviaron en 3 rondas, de modo que se obtuvieron 375 peticiones por servicio, tanto para el servicio de segmentación (/predict/segmentation) como para el de interpretabilidad con Grad-CAM

(/predict/xai). Las peticiones se enviaron de forma secuencial, es decir, una a la vez, tras una petición de calentamiento por servicio que no se contabilizó, y las imágenes se redujeron en el cliente a un máximo de 1024 px en su lado mayor (JPEG, calidad 0.92) antes de enviarse, igual que hace la aplicación; esta reducción se realizó antes de la medición, por lo que no está incluida en los tiempos. La medición se efectuó contra el servidor real de producción, que se ejecuta sobre CPU en Hugging Face Spaces, desde el navegador del usuario, el 11 de octubre de 2026.

Se registraron cuatro métricas: el tiempo de análisis (inferencia), que es el que emplea el servidor en procesar la imagen; la respuesta de la API, medida en el servidor; la red, que suma la subida de la imagen y la descarga del resultado; y el total percibido por el usuario, que corresponde a la suma de la respuesta de la API y la red. De cada una se calcularon la media con su desviación estándar (DE), la mediana, el percentil 95 (P95), el mínimo y el máximo. Por su parte, el tiempo de carga del modelo se registra una sola vez cada vez que arranca el servidor e incluye la lectura de los pesos y su carga en memoria; en el último arranque fue de 216 ms, por lo que se reporta como una única medición y no como un promedio. En cuanto al consumo de recursos del servicio de inferencia, el backend se ejecutó sobre el tier gratuito CPU Basic de Hugging Face Spaces (2 vCPU, 16 GB de RAM, 50 GB de disco no persistente); dado que el modelo ocupa solo 15.78 MB en disco (Tabla 59), su huella quedó muy por debajo de ese presupuesto, y durante las 375 peticiones del benchmark no se registraron errores, caídas ni degradación por falta de memoria.

Tabla 65. *Tiempos de respuesta de la aplicación en producción*

Servicio	Métrica	Media ± DE	Mediana	P95	Mín	Máx
Segmentación	Tiempo de análisis (inferencia)	115.7 ± 10.6	113.2	129.8	105.7	177.1
Segmentación	Respuesta de la API (servidor)	115.7 ± 10.6	113.2	129.8	105.7	177.1
Segmentación	Red (subida + descarga)	252.5 ± 124.3	234.9	330.1	218.1	2424.9
Segmentación	Total percibido por el usuario	368.2 ± 125.1	348.0	445.8	326.2	2551.9
Grad-CAM (XAI)	Tiempo de análisis (inferencia)	799.9 ± 38.2	790.1	864.2	745.1	1005.1
Grad-CAM (XAI)	Respuesta de la API (servidor)	800.0 ± 38.2	790.1	864.2	745.1	1005.1
Grad-CAM (XAI)	Red (subida + descarga)	462.5 ± 363.3	333.5	1277.9	221.2	3785.7
Grad-CAM (XAI)	Total percibido por el usuario	1262.4 ± 366.2	1144.4	2103.3	981.8	4564.6

Fuente: Elaboración propia, con base en el benchmark ejecutado sobre el servidor de producción (125 imágenes × 3 rondas, reducidas a 1024 px; inicio 11/10/2026, 16:44:19).

Como se observa en la Tabla 65, el servicio de segmentación respondió con rapidez y de forma estable: el tiempo de análisis tuvo una media de 115.7 ms (DE = 10.6 ms), una mediana de 113.2 ms y un P95 de 129.8 ms, y la respuesta de la API medida en el servidor fue prácticamente igual a este valor, lo que indica que el tiempo del servidor lo determina casi por completo la ejecución del modelo. Este valor es el que se reporta como latencia de inferencia en producción en la Tabla 59 y respalda la viabilidad de desplegar el modelo sobre CPU. La red aportó una media de 252.5 ms (mediana de 234.9 ms) con una variabilidad alta (DE = 124.3 ms y un máximo de 2424.9 ms) que depende de la conexión, de modo que el total percibido por el usuario fue de 368.2 ms de media (mediana de 348.0 ms) y, en el 95 % de las peticiones, no superó los 445.8 ms.

El servicio con Grad-CAM, que además de segmentar genera el mapa de calor, requirió más tiempo: el análisis tuvo una media de 799.9 ms (DE = 38.2 ms; mediana de 790.1 ms; P95 de 864.2 ms), es decir, casi siete veces el de la segmentación sola, aunque con una dispersión baja. La red tuvo una media de 462.5 ms (mediana de 333.5 ms) y una variabilidad mayor (DE = 363.3 ms y un máximo de 3785.7 ms), por lo que el total percibido fue de 1262.4 ms de media (mediana de 1144.4 ms) y, en el 95 % de las peticiones, no superó los 2103.3 ms. En conjunto, el diagnóstico con la máscara de segmentación se entrega en menos de medio segundo en la mayoría de los casos y, cuando se solicita el mapa de calor, en poco más de un segundo, tiempos que concuerdan con la valoración de los usuarios de la Tabla 64, donde los enunciados sobre el tiempo de generación del diagnóstico y sobre la ausencia de demoras en

el mapa de calor obtuvieron medias de 4.38 y 4.25, respectivamente. Cabe señalar que estos valores dependen de la carga del servidor y de la conexión desde la que se mide, por lo que deben interpretarse como una referencia del comportamiento observado en la fecha de la medición.

Por otra parte, el tiempo de carga de la interfaz web se evaluó con Google Lighthouse 13.5 sobre la aplicación desplegada en Render, mediante cinco ejecuciones por perfil (se reporta la mediana), con las condiciones de red simuladas por la herramienta (escritorio: 10 Mbps y 40 ms de latencia; móvil: 1.6 Mbps, 150 ms de latencia y CPU ralentizada 4×). En escritorio, el primer contenido se mostró a 1.00 s y el contenido principal (LCP) a 2.05 s, con un tiempo de bloqueo de 1.5 ms y sin desplazamientos de diseño (CLS = 0), valores dentro de los umbrales recomendados por Google; la puntuación de rendimiento fue de 86/100. En el perfil móvil, el LCP fue de 6.77 s y la puntuación de 62/100, limitados principalmente por el tamaño del paquete de JavaScript. Previamente, la conversión de la imagen de fondo de PNG a WebP redujo el peso de la página de 2.42 MB a 644 KB y el LCP móvil de 14.86 s a 6.77 s. Por su parte, la respuesta del servidor web (TTFB) fue de 15 ms, y los resultados en escritorio son coherentes con la valoración de 4.25 que los usuarios otorgaron al tiempo de carga de la aplicación en la Tabla 64.

4.3.6 Valoración de la usabilidad percibida en el uso de la aplicación

De igual manera, con base en las 8 respuestas que se recogieron hasta este corte, se calculó la media de cada uno de los trece enunciados que integran la Sección B del instrumento, es decir, la Usabilidad, que corresponde a las subcaracterísticas de capacidad para reconocer su idoneidad, capacidad de aprendizaje, operabilidad, protección contra errores de usuario, estética de la interfaz de usuario y accesibilidad. Los resultados se presentan en la Tabla 66.

Tabla 66. *Valoración de la usabilidad percibida por enunciado.*

Enunciado	Media	Nivel
Comprendí con facilidad que su propósito es detectar enfermedades en maíz.	4.50	Muy alto
La aplicación muestra de forma clara las funciones que ofrece.	4.38	Muy alto
Aprendí a utilizar la aplicación en poco tiempo y sin mucha ayuda externa.	4.75	Muy alto
El proceso para cargar una imagen y obtener el diagnóstico fue fácil de aprender.	4.88	Muy alto
La navegación entre secciones es sencilla e intuitiva.	4.38	Muy alto
Registrarme, iniciar sesión y gestionar mi cuenta resultaron fáciles.	4.50	Muy alto
Obtener el resultado del análisis requiere pocos pasos.	4.50	Muy alto
Cuando cometí un error, la aplicación me lo advirtió con claridad.	3.88	Alto
Pude corregir fácilmente los errores sin perder la información ya ingresada.	4.25	Muy alto
El diseño visual es agradable y ordenado.	4.62	Muy alto
La posibilidad de alternar entre tema claro y oscuro mejora mi comodidad.	4.50	Muy alto
Los textos, botones e íconos tienen tamaño y contraste adecuados.	4.38	Muy alto
Pude usar la aplicación cómodamente desde el navegador, sin instalar nada.	4.38	Muy alto

Fuente: Elaboración propia, con base en las 8 respuestas recopiladas hasta la fecha.

La media global de usabilidad percibida, considerando los trece enunciados, fue de 4.45 sobre 5, lo que corresponde a un nivel de valoración Muy alto según la escala que se definió en el apartado 4.3.2, donde el enunciado con la valoración más alta fue la facilidad para aprender a cargar una imagen y obtener el diagnóstico con 4.88, mientras que el único enunciado que se ubicó en el nivel Alto, es decir por debajo de Muy alto, fue la claridad de las advertencias ante errores del usuario con 3.88, lo que sugiere una oportunidad de mejora puntual en los mensajes de validación de la aplicación.

Además de la percepción recogida mediante la encuesta, se aplicó una prueba de usabilidad basada en tareas con 8 participantes, con el propósito de medir de forma objetiva la efectividad y la eficiencia del uso de la aplicación, que son medidas de la calidad en uso contempladas en la norma ISO/IEC 25010 y que se relacionan con las subcaracterísticas de operabilidad y capacidad de aprendizaje de la usabilidad. Cada participante resolvió cinco tareas sin recibir indicaciones sobre cómo realizarlas, a saber, subir una imagen de maíz (T1), ejecutar el análisis e indicar la enfermedad detectada (T2), abrir el mapa de calor e indicar la zona más afectada (T3), encontrar las recomendaciones de tratamiento (T4) y entrar al Historial para abrir un análisis anterior (T5). Una tarea se consideró exitosa cuando el participante la completó sin ayuda y se consideró error cuando no pudo completarla o desistió, de modo que la tasa de error es el complemento de la tasa de éxito, mientras que el tiempo se midió desde que se inició la tarea hasta que se terminó o el participante desistió, y se calculó únicamente con las tareas exitosas. Los resultados por tarea se presentan en la Tabla 67.

Tabla 67. *Resultados de la prueba de usabilidad por tarea (n = 8 participantes).*

Tarea	Completaron	Éxito (%)	Error (%)	Tiempo promedio (s)	Mediana (s)	Mín.–Máx. (s)
T1. Subir una imagen	8 de 8	100.0	0.0	12.6	13.0	9–16
T2. Ejecutar el análisis e indicar la enfermedad	6 de 8	75.0	25.0	45.7	44.0	40–53
T3. Abrir el mapa de calor e indicar la zona afectada	7 de 8	87.5	12.5	25.0	26.0	19–30
T4. Encontrar las recomendaciones de tratamiento	7 de 8	87.5	12.5	18.4	16.0	15–25
T5. Abrir un análisis anterior en el Historial	7 de 8	87.5	12.5	25.3	24.0	22–30
Total general	35 de 40	87.5	12.5	24.5	23.0	9–53

Fuente: Elaboración propia, con base en la prueba de usabilidad con 8 participantes. El tiempo se calcula solo con las tareas exitosas.

En conjunto se completaron 35 de las 40 tareas realizadas, lo que equivale a una tasa de éxito global de 87.5 % y a una tasa de error de 12.5 %, con un tiempo promedio de 24.5 s por tarea exitosa y una mediana de 23 s. La tarea más sencilla fue subir una imagen (T1), que los 8 participantes completaron en un promedio de 12.6 s, mientras que la identificación de la enfermedad (T2) obtuvo la tasa de éxito más baja, 75 %, y el mayor tiempo, 45.7 s, ya que dos participantes no lograron identificar la enfermedad y desistieron. Este tiempo es muy superior al total percibido de 368.2 ms para la segmentación que se reportó en la Tabla 65, lo que indica que la mayor parte de la duración de esta tarea corresponde a la lectura e interpretación del resultado por parte del usuario y no al procesamiento del sistema.

En las tareas T3, T4 y T5 se registró un solo fallo en cada una, es decir una tasa de éxito de 87.5 %, y las observaciones coinciden en que la dificultad estuvo en ubicar la función y no en comprender la tarea, ya que un participante no ubicó la zona afectada en el mapa de calor, otro no encontró las recomendaciones de tratamiento y otro no encontró el Historial, además de que dos participantes navegaron por varios menús antes de dar con una de esas funciones. Los intentos fallidos se prolongaron entre 61 y 86 s antes de que el participante desistiera, por lo que se identifica como mejora puntual

hacer más visibles esas secciones en la navegación. Como limitación, en dos sesiones la imagen analizada no contenía enfermedades, por lo que T3 y T4 se registraron como completadas, aunque no había zona afectada ni recomendaciones que aplicar, y el tamaño de la muestra, de 8 participantes, no permite generalizar estos porcentajes.

4.3.7 Percepciones de los usuarios y recomendaciones de mejora

En la Sección D del instrumento los 8 participantes respondieron la pregunta cerrada sobre la disposición a recomendar la aplicación, y la totalidad, es decir 8 de 8 que representan el 100 %, indicó que sí la recomendaría a otros agricultores o técnicos agrícolas.

Respecto a la pregunta abierta sobre los aspectos mejor valorados de la aplicación, los comentarios se concentraron principalmente en los mapas de calor, que se mencionaron en 2 respuestas, además del diseño visual de la interfaz con sus colores y su presentación general, la facilidad de uso y la capacidad de la aplicación para señalar de forma visual la zona afectada por la enfermedad en la imagen, y un participante destacó también la sección que recomienda cuándo utilizar un dron para la captura de imágenes.

En cuanto a las dificultades encontradas o las mejoras sugeridas, la mayoría de los comentarios coincidió en pedir que se cubra un mayor número de enfermedades del maíz, lo que se mencionó en 3 respuestas, y en obtener mejores recomendaciones agronómicas a partir del diagnóstico, mientras que un participante sugirió reordenar la interfaz de diagnóstico para hacerla más clara y otros dos indicaron no haber encontrado ninguna dificultad relevante durante el uso de la aplicación.

4.3.8 Discusión integral de los resultados de la evaluación de la aplicación web

En conjunto, los resultados de la evaluación de la aplicación web muestran un desempeño favorable en las dos características de calidad analizadas, aunque con matices que conviene señalar. En eficiencia de desempeño, la percepción de los 8 usuarios (media global de 4.43 sobre 5, nivel Muy alto) coincide con las mediciones objetivas: en producción, el diagnóstico con la máscara de segmentación se entregó con un total percibido de 368.2 ms de media y, cuando se solicitó el mapa de calor con Grad-CAM, de 1262.4 ms, tiempos que concuerdan con que los enunciados sobre el tiempo de generación del diagnóstico (4.38) y de los mapas de calor (4.25) se hayan valorado como Muy alto. El punto más débil fue la carga inicial de la interfaz en dispositivos móviles, con un LCP de 6.77 s y una puntuación de 62/100 en Lighthouse, frente a 2.05 s y 86/100 en escritorio.

En usabilidad, la media global de 4.45 sobre 5 (nivel Muy alto) y la facilidad para aprender a cargar una imagen y obtener el diagnóstico (4.88) se corresponden con la prueba por tareas, en la que se completaron 35 de las 40 tareas (tasa de éxito de 87.5 % y tasa de error de 12.5 %) con un tiempo

promedio de 24.5 s por tarea exitosa. Los fallos no se debieron a una mala comprensión de las tareas, sino a la dificultad para ubicar funciones concretas (el mapa de calor, las recomendaciones de tratamiento y el Historial) y para identificar la enfermedad en la tarea T2 (75 % de éxito). Esta dificultad se relaciona con el enunciado peor valorado de la encuesta, la claridad de las advertencias ante errores del usuario (3.88), de modo que tanto la percepción declarada como el desempeño observado señalan la misma oportunidad de mejora: hacer más visibles las secciones en la navegación y reforzar los mensajes de ayuda y de validación.

Por último, los 8 participantes (100 %) indicaron que recomendarían la aplicación, y valoraron sobre todo los mapas de calor y el diseño visual; las mejoras más solicitadas fueron cubrir un mayor número de enfermedades y obtener mejores recomendaciones agronómicas, lo que es coherente con que el modelo segmente únicamente tres enfermedades (tizón, roya y mancha blanca). Estos resultados deben interpretarse con cautela, pues la muestra es de 8 participantes y la recolección sigue en curso, en dos sesiones la imagen analizada no contenía enfermedades y los tiempos medidos dependen de la carga del servidor y de la conexión desde la que se mide. Con todo, la evidencia reunida respalda que la aplicación es eficiente y fácil de usar para el diagnóstico visual de enfermedades del maíz, y sirve de base para el despliegue en producción que se describe en la sección 4.4.

4.4 Despliegue de la aplicación

Una vez desarrollada y probada localmente, la aplicación se desplegó en un entorno de producción accesible desde Internet, con el fin de que pudiera ser utilizada por los usuarios finales sin necesidad de instalación alguna. Para ello se distribuyeron los tres componentes del sistema en plataformas en la nube, manteniendo la separación propia de la arquitectura cliente-servidor.

El frontend desarrollado en React se publicó en la plataforma Render, la cual se encarga de servir la interfaz al navegador del usuario. El backend en FastAPI se desplegó en Hugging Face Spaces, una plataforma que ofrece los recursos de cómputo necesarios para alojar el modelo de Deep Learning y ejecutar la inferencia. Este componente carga los pesos del modelo MiT-B0 y atiende las solicitudes del frontend. Finalmente, la base de datos PostgreSQL se alojó como un servicio gestionado en Supabase, encargado de almacenar de forma persistente los datos de los usuarios y el historial de análisis.

El flujo completo de funcionamiento en producción es el siguiente: el usuario accede a la aplicación desde su navegador, carga una imagen capturada con un dron, esta se envía al backend alojado en Hugging Face Spaces, donde el modelo realiza la segmentación y genera el mapa de calor, y el resultado, junto con el porcentaje de área afectada, se devuelve al frontend y se almacena en la base de datos. En la Tabla 68 se detallan los componentes del despliegue y la plataforma utilizada en cada caso, y en la Figura 34 se ilustra el flujo y la arquitectura completa del sistema.

Tabla 68. *Componentes del despliegue en producción.*

Componente	Plataforma	Función
Frontend (React)	Render	Entrega de la interfaz de usuario al navegador
Backend (FastAPI)	Hugging Face Spaces	Alojamiento del modelo e inferencia de segmentación
Base de datos (PostgreSQL)	Supabase	Almacenamiento de usuarios e historial de análisis

Fuente: Elaboración propia.

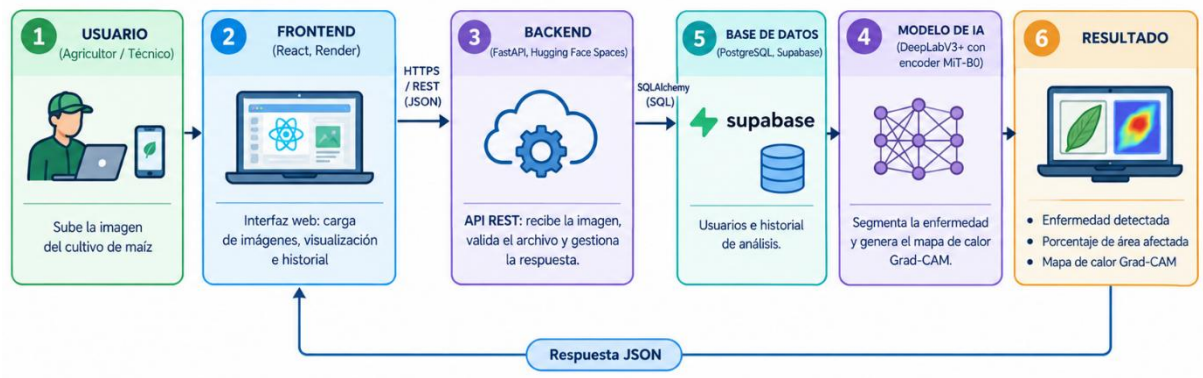


Figura 34. *Flujo y arquitectura de despliegue del sistema AgroScan.*

Fuente: Elaboración propia.

De esta manera, el sistema quedó completamente operativo en un entorno de producción, integrando el modelo de segmentación, la interfaz de usuario y el almacenamiento de datos en una única solución accesible en línea, lo que evidencia que es posible llevar un modelo de Deep Learning entrenado con imágenes de dron hasta un prototipo funcional y utilizable en el contexto de la agricultura de precisión.

En cuanto a la evidencia de funcionamiento y los tiempos de respuesta del sistema completo en producción, estos se documentan en la sección 4.3.5: la Tabla 65 reporta los tiempos medidos sobre 375 peticiones reales por servicio (tiempo de análisis, respuesta de la API, red y total percibido por el usuario), obtenidas contra el servidor ya desplegado, mientras que la Figura 34 ilustra el flujo íntegro desde la carga de la imagen hasta el almacenamiento del resultado.

Cabe señalar que los tres componentes operan bajo los planes gratuitos de sus respectivas plataformas, cada uno con restricciones propias de infraestructura. En Render, el servicio web gratuito se suspende tras 15 minutos sin tráfico entrante y tarda cerca de un minuto en reactivarse ante la siguiente solicitud, dentro de un límite de 750 horas de instancia gratuitas por mes y por espacio de trabajo. En Hugging Face Spaces, el backend corre sobre el tier gratuito CPU Basic (2 vCPU, 16 GB de RAM, 50 GB de disco no persistente); este entra en reposo tras 48 horas sin peticiones, por lo que el

disco se reinicia vacío en cada reactivación y el modelo (15.78 MB) debe descargarse nuevamente al arrancar el contenedor. En Supabase, la base de datos PostgreSQL gratuita está limitada a 500 MB de almacenamiento y se pausa automáticamente tras 7 días sin actividad, lo que exige una reactivación manual desde el panel de administración antes de volver a atender solicitudes. Estas restricciones no afectaron la medición de tiempos de respuesta reportada en la Tabla 65, realizada con los tres servicios ya activos, pero implican que, en un escenario de uso real con periodos prolongados de inactividad, el primer usuario puede experimentar una demora adicional de entre 30 y 60 segundos mientras el backend y el frontend se reactivan.

CONCLUSIONES

En cumplimiento del objetivo general, se desarrolló una aplicación web funcional que detecta automáticamente las enfermedades del maíz mediante una arquitectura híbrida de Deep Learning. En una misma solución en línea se integraron el modelo de segmentación, la interfaz de usuario y el almacenamiento de los diagnósticos, de modo que un modelo entrenado con imágenes de dron terminó convertido en una herramienta real y utilizable en campo.

En cumplimiento del primer objetivo específico, se definieron los fundamentos teóricos de las enfermedades que afectan al maíz y de las técnicas de detección basadas en aprendizaje profundo (Capítulo II): se revisaron la vulnerabilidad del cultivo frente al tizón, la roya y la mancha blanca, los enfoques híbridos CNN-Transformer y las arquitecturas livianas de Deep Learning, el papel de los UAV en la agricultura de precisión, el modelo de calidad ISO/IEC 25010 y los trabajos relacionados, lo que sustentó la posterior elección de una arquitectura híbrida liviana y de un esquema de evaluación alineado con dicha norma.

En cumplimiento del segundo objetivo específico, el dataset se construyó siguiendo la metodología CRISP-DM. Se partió de imágenes recolectadas con dron y se complementaron con datasets públicos. Todas se anotaron en Roboflow, se preprocesaron y se sometieron a aumento de datos. El desbalance entre clases se atendió mediante aumento de datos en Roboflow, lo que dejó un conjunto de entrenamiento más amplio y variado.

Se integró un modelo híbrido y liviano, DeepLabV3+ con encoder MiT-B0. Se lo eligió frente a otras variantes por su mejor equilibrio entre precisión y costo computacional. Además, se sumó la técnica de IA explicable Seg-Grad-CAM, que genera mapas de calor para respaldar cada predicción y darle más confianza al usuario.

En cumplimiento del tercer objetivo específico, los resultados sobre validación fueron sólidos: mIoU de 0.8182, exactitud de píxeles de 0.9767, F1-score macro de 0.8972 y mAP de 0.9468, con un mIoU de 0.7593 sobre el conjunto de prueba independiente. El desempeño se mantuvo equilibrado entre las cuatro clases en validación, pero la clase roya mostró una caída notable al pasar a prueba (IoU de 0.7463 a 0.5853), la mayor de las cuatro clases, lo que señala el principal margen de mejora: reforzar la representación de esta enfermedad en el conjunto de entrenamiento.

Asimismo, en cumplimiento del segundo objetivo específico, la aplicación se construyó con Scrum, sobre una arquitectura cliente-servidor de tres capas: frontend en React, backend en FastAPI y base de datos PostgreSQL. Quedó organizada en tres módulos: autenticación, análisis y diagnóstico de

imágenes, e historial de resultados. Se desplegó en un entorno de producción, lo que confirmó que el modelo entrenado puede ponerse en manos de un usuario final sin conocimientos técnicos.

Como parte del tercer objetivo específico, para evaluarla se usó un instrumento estructurado alineado con la norma ISO/IEC 25010. Con las respuestas recogidas hasta este corte, la eficiencia de desempeño percibida llegó a una media de 4.43 sobre 5 y la usabilidad a 4.45, ambas en el nivel Muy alto, y todos los participantes dijeron que la recomendarían. Son resultados preliminares, porque la recolección sigue abierta, pero apuntan a una buena percepción de la calidad en uso.

En cuanto a las herramientas empleadas, el proyecto combinó un stack de aprendizaje profundo (Python, PyTorch, Roboflow) con tecnologías web (React, FastAPI, PostgreSQL) y servicios en la nube (Google Colab, Render, Supabase, Hugging Face Spaces), lo que permitió llevar un modelo de Deep Learning entrenado con imágenes de dron hasta un prototipo funcional de extremo a extremo. La experiencia del desarrollo confirmó que la gestión ágil con Scrum facilitó iterar sobre los requisitos y priorizar funcionalidades, y que trabajar con datos reales de campo heterogéneos en iluminación, altura de vuelo y fuente, exige un esfuerzo de preparación de datos comparable al del propio modelado.

RECOMENDACIONES

A partir de los hallazgos obtenidos y de las limitaciones identificadas a lo largo del trabajo, se plantean las siguientes recomendaciones orientadas a fortalecer y dar continuidad a la solución desarrollada:

Conviene completar la evaluación con usuarios y ampliar la muestra más allá de las respuestas actuales. Con más encuestas, los valores de eficiencia y usabilidad ganarían respaldo estadístico y dejarían de ser preliminares.

La roya es la clase que más hay que reforzar: fue la de menor recall (0.8503 en validación y 0.6625 en prueba) y la de mayor margen de mejora. Para eso conviene sumar más imágenes de esta enfermedad, sobre todo en sus etapas leves, y probar estrategias de aumento de datos y balanceo pensadas para esa clase.

También vale la pena ampliar y diversificar el dataset: nuevas imágenes en distintas condiciones de campo, etapas de crecimiento, zonas geográficas y niveles de luz, e incluso otras enfermedades. Con eso el modelo generalizaría mejor ante escenarios reales más variados.

La retroalimentación al usuario se puede mejorar, en especial los mensajes y advertencias ante errores, que fue lo peor valorado dentro de la usabilidad. Unas guías más claras ayudarían a que personas sin conocimientos técnicos aprovechen mejor la herramienta.

Como la arquitectura es liviana, se podría evaluar una versión móvil o con funcionamiento sin conexión. Muchos usuarios del sector agrícola trabajan en zonas rurales donde la conectividad es limitada.

Por último, sería útil montar un mecanismo de reentrenamiento continuo que aproveche los diagnósticos generados durante el uso real. Así el modelo se actualizaría cada cierto tiempo, se podría vigilar su desempeño y mantendría su precisión ante nuevos patrones o condiciones.

BIBLIOGRAFÍA

[1] L. Stefanía and M. López, "Análisis socioeconómico de los productores de maíz de la parroquia La Esperanza, Imbabura.," Ibarra, 2023. Accessed: Nov. 04, 2025. [Online]. Available: https://repositorio.utn.edu.ec/bitstream/123456789/16255/2/03%20AGP%20431%20TRABAJO%20DE%20GRADO.pdf

[2] A. Pfordt and S. Paulus, "A review on detection and differentiation of maize diseases and pests by imaging sensors," Feb. 01, 2025, *Springer Science and Business Media Deutschland GmbH*. doi: 10.1007/s41348-024-01019-4.

[3] H. Zhou *et al.*, "Maize Leaf Disease Recognition Based on Improved Convolutional Neural Network ShuffleNetV2," *Plants*, vol. 13, no. 12, p. 1621, Jun. 2024, doi: 10.3390/plants13121621.

[4] D. Telenko, M. Chilvers, A. Robertson, A. Tenuta, and D. Smith, "Will a Second Fungicide be Worth the Cost for Tar Spot Management?," Aug. 2021, doi: 10.31274/CPN-20210820-1.

[5] T. Mizik, "How can precision farming work on a small scale? A systematic literature review," Feb. 01, 2023, *Springer*. doi: 10.1007/s11119-022-09934-y.

[6] Y. Zhang, S. Wa, Y. Liu, X. Zhou, P. Sun, and Q. Ma, "High-Accuracy Detection of Maize Leaf Diseases CNN Based on Multi-Pathway Activation Function Module," *Remote Sens. (Basel).*, vol. 13, no. 21, p. 4218, Oct. 2021, doi: 10.3390/rs13214218.

[7] "Corn Disease Drone Images." Accessed: Nov. 18, 2025. [Online]. Available: https://www.kaggle.com/datasets/alexanderyevchenko/corn-disease-drone-images

[8] P. Zhang, X. Sun, D. Zhang, Y. Yang, and Z. Wang, "Lightweight Deep Learning Models for High-Precision Rice Seedling Segmentation from UAV-Based Multispectral Images," *Plant Phenomics*, vol. 5, p. 0123, Jan. 2023, doi: 10.34133/PLANTPHENOMICS.0123.

[9] K. Sharma and S. K. Shivandu, "Integrating artificial intelligence and Internet of Things (IoT) for enhanced crop monitoring and management in precision agriculture," Jan. 01, 2024, *KeAi Communications Co.* doi: 10.1016/j.sintl.2024.100292.

[10] A. Rengifo, S. Boza, and M. D. C. Pérez, "Assessment for the Insertion of Agroecological Female Farmers in the Provision of Restaurants," *Revista Iberoamericana de Viticultura Agroindustria y Ruralidad*, vol. 9, no. 26, pp. 191–210, May 2022, doi: 10.35588/rivar.v9i26.5579.

[11] I. Cisternas, I. Velásquez, A. Caro, and A. Rodríguez, "Systematic literature review of implementations of precision agriculture," *Comput. Electron. Agric.*, vol. 176, p. 105626, Sep. 2020, doi: 10.1016/J.COMPAG.2020.105626.

[12] W. Zhao *et al.*, "Effects of water stress on photosynthesis, yield, and water use efficiency in winter wheat," *Water (Switzerland)*, vol. 12, no. 8, Aug. 2020, doi: 10.3390/W12082127.

[13] M. C. De Cól *et al.*, "Quantitative summary of nine years of fungicide efficacy data for managing white spot and protecting yield in maize," *Crop Protection*, vol. 197, p. 107339, Nov. 2025, doi: 10.1016/J.CROPRO.2025.107339.

[14] Víctor A. González Huitrón, Jesús A. Medrano Hermosillo, Abraham E. Rodríguez Mata, José A. Almaráz Damián, and Pedro Celestino Ramírez, “Detección de enfermedades foliares en maíz mediante aprendizaje profundo,” pp. 40–44, 2025, Accessed: Oct. 30, 2026. [Online]. Available: https://www.researchgate.net/publication/387877416_Deteccion_de_enfermedades_foliares_en_maiz_mediante_aprendizaje_profundo

[15] X. Song *et al.*, “Agricultural Image Processing: Challenges, Advances, and Future Trends,” Aug. 01, 2025, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/app15169206.

[16] S. Rasti, C. J. Bleakley, N. M. Holden, R. Whetton, D. Langton, and G. O’Hare, “A survey of high resolution image processing techniques for cereal crop growth monitoring,” *Information Processing in Agriculture*, vol. 9, no. 2, pp. 300–315, Jun. 2022, doi: 10.1016/J.INPA.2021.02.005.

[17] M. R. Tonmoy, Md. M. Hossain, N. Dey, and M. F. Mridha, “MobilePlantViT: A Mobile-friendly Hybrid ViT for Generalized Plant Disease Image Classification,” Mar. 2025, [Online]. Available: http://arxiv.org/abs/2503.16628

[18] “Detección de enfermedades de cultivos impulsada por IA: aprendizaje profundo y vehículos aéreos no tripulados.” Accessed: Dec. 12, 2025. [Online]. Available: https://flypix.ai/es/crop-disease-detection/

[19] R. Shao and X. J. Bi, “Transformers Meet Small Datasets,” *IEEE Access*, vol. 10, pp. 118454–118464, 2022, doi: 10.1109/ACCESS.2022.3221138.

[20] Z. Dai, H. Liu, Q. V. Le, and M. Tan, “CoAtNet: Marrying Convolution and Attention for All Data Sizes,” Sep. 2021, [Online]. Available: http://arxiv.org/abs/2106.04803

[21] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” Apr. 2017, [Online]. Available: http://arxiv.org/abs/1704.04861

[22] H. Amin, A. Darwish, A. E. Hassanien, and M. Soliman, “End-to-End Deep Learning Model for Corn Leaf Disease Classification,” *IEEE Access*, vol. 10, pp. 31103–31115, 2022, doi: 10.1109/ACCESS.2022.3159678.

[23] D. Kapetas *et al.*, “Multispectral AI-driven imaging for detection of downy mildew and gray mold in grapevines,” *Smart Agricultural Technology*, vol. 14, Aug. 2026, doi: 10.1016/j.atech.2026.101990.

[24] K. Gopalan, S. Srinivasan, . P., M. Singh, S. K. Mathivanan, and U. Moorthy, “Corn leaf disease diagnosis: enhancing accuracy with resnet152 and grad-cam for explainable AI,” *BMC Plant Biol.*, vol. 25, no. 1, p. 440, Apr. 2025, doi: 10.1186/s12870-025-06386-0.

[25] K. Vinogradova, A. Dibrov, and G. Myers, “Towards Interpretable Semantic Segmentation via Gradient-weighted Class Activation Mapping,” Feb. 2020, doi: 10.1609/aaai.v34i10.7244.

[26] C. Lu, E. Nnadozie, M. P. Camenzind, Y. Hu, and K. Yu, “Maize plant detection using UAV-based RGB imaging and YOLOv5,” *Front. Plant Sci.*, vol. 14, Jan. 2024, doi: 10.3389/fpls.2023.1274813.

[27] J. Cuaran and J. Leon, “Crop Monitoring using Unmanned Aerial Vehicles: A Review,” *Agricultural Reviews*, no. Of, Mar. 2021, doi: 10.18805/ag.r-180.

[28] A. Rejeb, A. Abdollahi, K. Rejeb, and H. Treiblmaier, “Drones in agriculture: A review and bibliometric analysis,” *Comput. Electron. Agric.*, vol. 198, p. 107017, Jul. 2022, doi: 10.1016/J.COMPAG.2022.107017.

[29] “ISO/IEC 25010:2023 - Ingeniería de sistemas y software — Requisitos y evaluación de la calidad de sistemas y software (SQuaRE) — Modelo de calidad del producto.” Accessed: Jan. 28, 2026. [Online]. Available: https://www.iso.org/standard/78176.html

[30] H. Rojas, R. Renteria, V. M. Duran, Y. T. Gutiérrez, M. J. I. Cabrera, and A. Aminuddin, “Mapping the Evolution and Future Directions of ISO/IEC 25010: A Bibliometric and Thematic Analysis,” *Engineering, Technology & Applied Science Research*, vol. 15, no. 5, pp. 27530–27541, Oct. 2025, doi: 10.48084/etasr.11772.

[31] G. T. Askale, A. B. Yibel, B. M. Taye, and G. D. Wubneh, “Mobile based deep CNN model for maize leaf disease detection and classification,” Dec. 01, 2025, *BioMed Central Ltd*. doi: 10.1186/s13007-025-01386-5.

[32] D. T. Weldeslasie, M. Y. Mekonen, A. M. Abebe, and K. T. Desta, “A statistically validated stacking ensemble of CNNs and vision transformer for robust maize disease classification,” *Discover Artificial Intelligence*, vol. 5, no. 1, p. 284, Oct. 2025, doi: 10.1007/s44163-025-00548-7.

[33] Md. A. Haque *et al.*, “A lightweight convolutional neural network for recognition of severity stages of maydis leaf blight disease of maize,” *Front. Plant Sci.*, vol. 13, Dec. 2022, doi: 10.3389/fpls.2022.1077568.

[34] P. Theerthagiri, A. U. Ruby, J. G. C. Chandran, T. H. Sardar, and A. Shafeeq B. M, “Deep SqueezeNet learning model for diagnosis and prediction of maize leaf diseases,” *J. Big Data*, vol. 11, no. 1, Dec. 2024, doi: 10.1186/s40537-024-00972-z.

[35] E. L. Stewart *et al.*, “Quantitative Phenotyping of Northern Leaf Blight in UAV Images Using Deep Learning,” *Remote Sens. (Basel).*, vol. 11, no. 19, Oct. 2019, doi: 10.3390/rs11192209.

[36] F. Ciaglia, F. S. Zuppichini, P. Guerrie, M. McQuade, and J. Solawetz, “Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark,” Nov. 2022, [Online]. Available: http://arxiv.org/abs/2211.13523

[37] S. Chen *et al.*, “A Weakly Supervised Approach for Disease Segmentation of Maize Northern Leaf Blight from UAV Images,” *Drones*, vol. 7, no. 3, Mar. 2023, doi: 10.3390/drones7030173.

[38] B. Li, J. Zhou, F. Gou, and J. Wu, “TransRNetFuse: a highly accurate and precise boundary FCN-transformer feature integration for medical image segmentation,” *Complex and Intelligent Systems*, vol. 11, no. 5, May 2025, doi: 10.1007/s40747-025-01847-3.

[39] I. Agustian, I. Ardiyanto, and S. Wibirama, “Efficient crop–weed semantic segmentation via knowledge distillation from DeepLabV3+ to Fast-SCNN,” *Green Technologies and Sustainability*, vol. 4, no. 2, Apr. 2026, doi: 10.1016/j.grets.2026.100357.

[40] S. Soomro, R. Maskeliunas, R. Damasevicius, and A. Kurti, “SilvaFormer: Transformer-Based Segmentation for Forest Health Monitoring in UAV Imagery,” *IEEE Access*, 2026, doi: 10.1109/ACCESS.2026.3712179.

[41] Y. Yang, C. Wang, Q. Zhao, G. Li, and H. Zang, “SE-SWIN UNET FOR IMAGE SEGMENTATION OF MAJOR MAIZE FOLIAR DISEASES,” *Engenharia Agricola*, vol. 44, 2024, doi: 10.1590/1809-4430-Eng.Agric.v44e20230097/2024.

[42] Y. Gong, C. Yan, M. Xiao, M. Lu, and H. Gao, “Data augmentation method for light guide plate based on improved CycleGAN,” *Optoelectron. Lett.*, vol. 21, no. 9, pp. 555–561, Sep. 2025, doi: 10.1007/s11801-025-4092-y.

[43] S. Zhang, C. Xu, Y. Zhao, and T. Zhang, “TDAVM-UNet: task-driven attention VM-UNet for crop disease detection from UAV imagery,” *Front. Plant Sci.*, vol. 17, Jun. 2026, doi: 10.3389/fpls.2026.1855733.

ANEXOS

Código de entrenamiento

https://colab.research.google.com/drive/1Y-CIxG1fyIusKXPSDERJJPRkuryho303?usp=sharing

Dataset

https://drive.google.com/file/d/1TUrIr-zuStQbGPmFJ3MQ4k_uMXwqz8ot/view?usp=sharing

Aplicación web

Detección de Enfermedades en maíz